Matematyka Dyskretna

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

UMCS

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt „Programowa i strukturalna reforma systemu kształcenia na Wydziale Mat-Fiz-Inf”.
Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

Człowiek-najlepsza inwestycja

Uniwersytet Marii Curie-Skłodowskiej
Wydział Matematyki, Fizyki i Informatyki
Instytut Informatyki

Matematyka Dyskretna

Andrzej Krajka

Moim wspaniałym dzieciom
Asi i Tomkowi

Lublin 2011

Instytut Informatyki UMCS
Lublin 2011

Andrzej Krajka
Matematyka Dyskretna

Recenzent: Ryszard Smarzewski

Opracowanie techniczne: Marcin Denkowski
Projekt okładki: Agnieszka Kuśmierska

Praca współfinansowana ze środków Unii Europejskiej w ramach
Europejskiego Funduszu Społecznego

Publikacja bezpłatna dostępna on-line na stronach
Instytutu Informatyki UMCS: informatyka.umcs.lublin.pl.

Wydawca
Uniwersytet Marii Curie-Skłodowskiej w Lublinie
Instytut Informatyki
pl. Marii Curie-Skłodowskiej 1, 20-031 Lublin
Redaktor serii: prof. dr hab. Paweł Mikołajczak
www: informatyka.umcs.lublin.pl
email: dyrii@hektor.umcs.lublin.pl

Druk
ESUS Agencja Reklamowo-Wydawnicza Tomasz Przybylak
ul. Ratajczaka 26/8
61-815 Poznań
www: www.esus.pl

ISBN: 978-83-62773-05-3

Wstęp

Matematyka dyskretna jest to zbiorcza nazwa różnych działów matematyki, zajmujących się badaniem struktur nieciągłych czyli skończonych lub co najwyżej przeliczalnych. Matematyka dyskretna stała się popularna w ostatnich latach dzięki zastosowaniom w informatyce, która w sposób naturalny zajmuje się jedynie strukturami skończonymi (skończona reprezentacja liczb, skończona ilość operacji w jednostkach czasu komputera - taktowanie zegara).

Skrypt ten powstał na podstawie wykładów, które prowadziłem na kierunkach informatycznych Uniwersytetu im. Marii Curie-Skłodowskiej w Lublinie. Z założenia jest poszerzeniem wykładu o niektóre elementy, które z powodu ograniczeń czasowych nie mogły być w sposób kompletny wyłożone. Tak więc niektóre twierdzenia, podawane na wykładzie bez dowodów, tutaj są wprowadzone z dowodami i konieczną do tych dowodów teorią.

Nie zakładam jednak, że ta książka jest pełną monografią wszystkich działów matematyki dyskretnej. Ponieważ w planie zajęć studenci mają takie przedmioty jak podstawy rachunku prawdopodobieństwa, wstęp do logiki i teorii mnogości czy też elementy logiki i arytmetyki komputerów, więc poważnie zostały okrojone, a nawet zlikwidowane, takie działy matematyki dyskretnej jak algebra, logika, teoria mnogości, rachunek prawdopodobieństwa i kombinatoryka. Co więcej, ze względu na plan studiów, w tym skrypcie zakładam znajomość niektórych pojęć z tych działów (rozdział 7.1 należy traktować jako raczej przypomnienie a nie formalne wprowadzenie prawdopodobieństwa). Z drugiej strony, ponieważ studenci na drugim roku studiów mają przedmiot o nazwie teoria grafów, dlatego w ramach kursu - Matematyka Dyskretna teoria grafów została zredukowana do podstawowych pojęć. Traktujemy ją tutaj jako wprowadzenie do wspomnianego wyżej przedmiotu, chociaż i tak w tym skrypcie zagadnienia związane z grafami są znacznie bardziej rozbudowane w porównaniu z wykładem. Również duża ilość przedmiotów takich jak podstawy programowania spowodowała usunięcie z wykładu elementów dotyczących sprawdzania poprawności logicznej programów.

To co zostało to głównie problemy związane z rekurencjami, teoria liczb,

sumy i metody różnicowe, funkcje całkowitoliczbowe czy podstawy teorii grafów. Również wprowadzam, w dość okrojonym stopniu, rozdział dotyczący kombinatoryki, algorytmów sortowania i *haszowania.* Przy tworzeniu wykładu pojawił się problem wprowadzenia pojęcia ilorazu różnicowego. Najbardziej naturalne wprowadzenie polega na zastosowaniu analogii do różniczkowania i całkowania, jednak o ile pochodna jest obecna w programie szkół średnich, o tyle pojęcie całki w większości szkół w chwili obecnej jest usunięte. W wykładzie utrzymałem analogię z całkami, ponieważ pojęcie całki na I roku studiów jest równolegle wprowadzane na analizie jak również używane (nieformalnie) na niektórych przedmiotach fizycznych. Zdaję sobie jednak sprawę z tego, że rozdział 2.7.3 można skonstruować bez analogii z całką

Wykład starałem się konstruować na zasadzie od rozważenia pewnych przykładów do formułowania prawidłowości. Chociaż zarówno podręcznik jak i wykład mógłby mieć logicznie spójniejszą formę (na przykład wszystkie metody rozwiązywania równań rekurencyjnych zgromadzone w jednym miejscu), myślę jednak, że stałoby się to ze szkodą dla dydaktyki.

Algorytmy podawane są albo opisowo, albo w zbliżonym do Pascala pseudokodzie. W szczególności, w porównaniu z Pascalem często zamiast **begin** i **end** używać będziemy nawiasów { i } (również używając tych nawiasów do komentarzy) lub podobnych do Pythona wcięć, zwiększając przez to czytelność programu. Opuszczać też będziemy słówko **var** w deklaracjach a do operacji łączenia łańcuchów używać będziemy operatora &.

Oznaczenia

$x \wedge y$ - $\min\{x, y\}$
$x \vee y$ - $\max\{x, y\}$
$f(n) \asymp g(n)$ - $f(n)$ i $g(n)$ mają ten sam współczynnik przyrostu, tzn. istnieje taka stała C, że $|f(n)| \leq C|g(n)|$ i $|g(n)| \leq C|f(n)|$
$f(n) \sim g(n)$ - $f(n)$ jest asymptotyczna do $g(n)$ jeśli $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$
$f(n) \prec g(n)$ - $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$
$\overline{\overline{X}}$ - moc zbioru X (dla skończonych zbiorów jest to ilość elementów zbioru X)
$m \perp n$ - liczby m i n są względnie pierwsze, tzn. $NWD(m, n) = 1$
$P \perp Q$ - wielomiany P i Q są względnie pierwsze, tzn. $NWD(P, Q) = 1$
$a \equiv b \bmod m$ - a przystaje do b modulo m jeśli $(a \ mod \ m) = (b \ mod \ m)$ lub równoważnie $m|(a - b)$
$\lfloor x \rfloor$ - funkcja "podłoga" równa $\max\{c \in Z : c \leq x\}$
$\lceil x \rceil$ - funkcja "sufit" równa $\min\{c \in Z : c \geq x\}$
$\{x\}$ - funkcja "część ułamkowa" równa $x - \lfloor x \rfloor$
$x^{\underline{m}}$ - m ta dolna silnia od x definiowana jako

$$x^{\underline{m}} = \begin{cases} x(x-1)(x-2)...(x-m+1), & \text{dla } m > 0, \\ 1, & \text{dla } m = 0, \\ \frac{1}{(x+1)(x+2)(x+3)...(x+(-m))}, & \text{dla } m < 0. \end{cases}$$

$\binom{r}{k}$ - uogólniony symbol Newtona, dla $k \in \mathcal{N}$ równy $\frac{r^{\underline{k}}}{k!}$ podczas gdy dla $k \in \mathcal{Z}_-$ $\binom{r}{k} = 0$
$m|n$ - m jest dzielnikiem n tzn. istnieje liczba całkowita k taka, że $n = km$
$P|Q$ - wielomian P jest dzielnikiem Q tzn. istnieje wielomian R taki, że $Q = PR$
$m||n$ - m jest dokładnym dzielnikiem n tzn. istnieje liczba pierwsza p i liczba r takie, że $m = p^r|n$ i $p^{r+1} \not| n$
$x \bmod y$ - operacja modulo równa $x \bmod y = x - y\lfloor \frac{x}{y} \rfloor$ dla $y \neq 0$ oraz $x \bmod 0 = x$
$A_{min}, A_{sr}, A_{max}, A^i_{min}, A^i_{sr}, A^i_{max}$ - ilość przesunięć rekordów potrzebnych przy dodawaniu nowego rekordu minimalnie, przeciętnie i w najgorszym

przypadku. Litery z indeksem i dotyczą przesuwania nie rekordów ale indeksów

$A(z)$ - ($[a_0, ..]$) funkcja tworząca ciągu to funkcja $A(z) = [a_o, a_1, a_2, ...] = a_o + a_1 z + a_2 z^2 + ... = \sum_{k \geq 0} a_k z^k$, n-ty wyraz funkcji yworzącej równy a_n zapisujemy jako $z^n(A(z))$

B_n - bukiet jest grafem zawierającym jeden wierzchołek i n pętli w tym wierzchołku

$\mathcal{C}$ - zbiór liczb zespolonych

C_n - graf n-cykl to graf zawierający n wierzchołków $V(\mathcal{C}_n) = \{v_1, v_2, ..., v_n\}$ i dokładnie n krawędzi $E(\mathcal{C}_n) = \{\{v_1, v_2\}, \{v_2, v_3\}, ..., \{v_{n-1}, v_n\}, \{v_n, v_1\}\}$.

C_n^k - ilość kombinacji bez powtórzeń k elementów ze zbioru n elementowego, $C_n^k = \binom{n}{k}$

$\overline{C}_n^k$ - ilość kombinacji z powtórzeniami k elementów ze zbioru n elementowego, $\overline{C}_n^k = \binom{n+k-1}{k}$

$(c_n ... c_1 c_0, c_{-1} ... c_{-k})_p$ - zapis liczby cyframi w układzie o podstawie liczenia p równy $\sum_{i=-k}^{n} c_i p^i$

$Df(x), f'(x), \frac{df(x)}{dx}$ - operator różniczkowania równy $\lim_{h \to 0} \frac{f(x+h)-f(x)}{h}$

D_n - dipol jest grafem zawierającym dwa wierzchołki i n krawędzi łączących te wierzchołki

$D_n^{(q)}$ - ciąg rekurencyjny wspomagający rozwiązanie problemu Flawiusza: $D_0^{(q)} = 1, \quad D_n^{(q)} = \lceil \frac{q}{q-1} D_{n-1}^{(q)} \rceil, n \geq 1$

$\deg P(x)$ - stopień wielomianu $P(x)$ - najmniejsza liczba naturalna n taka, że $\triangle^n P(x) \equiv 0$.

$\deg_{\mathcal{G}}(v)$ - stopień wierzchołka v w grafie $\mathcal{G}$, jest to ilość krawędzi do których należy v

$\triangle f(x)$ - operator różnicowy równy $f(x+1) - f(x)$

E - operator przesunięcia, $Ef(x) = f(x+1)$

$E(\mathcal{G})$ - zbiór krawędzi grafu $\mathcal{G}$

EX - wartość oczekiwana zmiennej losowej X

$\mathcal{G}_\emptyset$ - graf pusty $(\emptyset, \emptyset)$

$\mathcal{G}(V_1, V_2)$ - graf dwudzielny jest to graf którego zbiór wierzchołków można podzielić na dwa podzbiory takie, że żadne dwa wierzchołki z tego samego podzbioru nie są połączone krawędzią

H_n - n-ta liczba harmoniczna $H_n = \sum_{k=1}^{n} \frac{1}{k}$

$I(A)$ - indykator zdarzenia A, funkcja równa 1 dla $\omega \in A$, i równa 0 dla $\omega \notin A$

$I_{\mathcal{G}}$ - lista krawędzi grafu $\mathcal{G}$, lista na której bezpośrednio podajemy wszystkie krawędzie grafu

$indeg_{\mathcal{G}}(v)$ - stopień wchodzący wierzchołka v w grafie skierowanym $\mathcal{G}$, jest to ilość krawędzi które kończą się v

$J_k(n)$ rekurencja Flawiusza

K_n - graf kompletny to graf prosty składający się z n wierzchołków i krawędzi łączących każde dwa wierzchołki

K_{r_1,r_2} - pełny graf dwudzielny jest to graf którego zbiór wierzchołków można podzielić na dwa podzbiory liczebności odpowiednio r_1 oraz r_2 takie, że żadne dwa wierzchołki z tego samego podzbioru nie są połączone krawędzią a zbiór krawędzi zawiera wszystkie krawędzie łączące wierzchołki jednego podzbioru z wierzchołkami drugiego podzbioru. Analogicznie definiujemy grafy trójdzielne K_{r_1,r_2,r_3} itd.

$L_{\mathcal{G}}$ - lista incydencji grafu $\mathcal{G}$. Lista dla każdego wierzchołka v, w której przechowujemy zbiór wierzchołków połączonych krawędzią z v

l_n - liczba obszarów w problemie pizzy $l_n = 1 + \frac{n(n+1)}{2}, n \geq 0$

$LCG(M, a, b, \alpha)$ - liniowy generator liczb pseudolosowych Lehmera tworzony wg. wzoru rekurencyjnego $X_n = (aX_{n-1} + b) \mod M$, $n > 1$, z zainicjowaną w jakiś sposób wartością X_o

$M(p)$ - liczba Mersenne'a (wynosi $2^p - 1$).

$M_{\mathcal{G}}$ - macierz sąsiedztwa $[m_{i,j}]$ grafu $\mathcal{G}$, $m_{i,j}$ oznacza ilość krawędzi grafu $\mathcal{G}$ łączących i-ty i j-ty wierzchołek

$\mathcal{N}$ - zbiór liczb naturalnych, o ile nie zostanie to wyraźnie stwierdzone zawsze z 0

$N(\alpha, n)$ - ilość elementów w widmie $Spec(\alpha)$ nie większych niż n

$NW_{min}, NW_{sr}, NW_{max}, NW^i_{min}, NW^i_{sr}, NW^i_{max}$ - ilość przejrzanych rekordów potrzebnych do znalezienia poszukiwanego rekordu (minimalna, przeciętna i najgorsza) gdzy tego rekordu nie znaleźliśmy. Litery z indeksem i dotyczą przesuwania nie rekordów ale indeksów

$NWD(P, Q)$ - największy wspólny dzielnik wielomianów P i Q, zakładamy, że zawsze najwyższy współczynnik wielomianu $NWD(P, Q)$ wynosi 1

$NWD(m, n)$ - największy wspólny dzielnik licz m i n równy $\max\{k : k|m \wedge k|n$

$NWW(m, n)$ - najmniejsza wspólna wielokrotność liczb m i n równa $\min\{k : m|k \wedge n|k\}$

$O(f(n))$ - symbol O, $g(n) = O(f(n))$ gdy $\exists_C \forall_{n \in \mathcal{N}} |f(n)| \leq C|g(n)|$

$o(f(n))$ - symbol o, $g(n) = o(f(n))$ gdy $\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$

$outdeg_{\mathcal{G}}(v)$ - stopień wychodzący wierzchołka v w grafie skierowanym $\mathcal{G}$, jest to ilość krawędzi zaczynających się od v

$P(A|B)$ - $(P[A|B])$ prawdopodobieństwo zajścia zdarzenia A pod warunkiem, że wiemy iż zaszło zdarzenie B

P_n - ilość permutacji bez powtórzeń $P_n = n!$

$P_n^{n_1,n_2,...,n_k}$ - ilość permutacji z powtórzeniami $(n_1, n_2, ..., n_k)$ gdzie $n = n_1 + n_2 + ... + n_k$ wtedy $P_n^{n_1,n_2,...,n_k} = \frac{n!}{n_1!n_2!...n_k!}$,

$P^{\lambda_1,\lambda_2,...,\lambda_n}$ - ilość podziałów zbioru $\{1, 2, ..., n\}$ na λ_i zbiorów liczebności $i, 1 \leq i \leq n$ przy czym $n = \lambda_1 + \lambda_2 + ... + \lambda_n$ wynosi $\frac{n!}{\lambda_1!\lambda_2!...\lambda_n!(1!)^{\lambda_1}(2!)^{\lambda_2}...(n!)^{\lambda_n}}$

$P(n,k)$ - ilość przedstawień liczby n jako suma k składników

Pet - to graf Petersena jest to graf skonstruowany z dwóch rozłącznych 5-cykli $\{u_0,u_1,u_2,u_3,u_4\}$ oraz $\{v_0,v_1,v_2,v_3,v_4\}$ połączone miedzy sobą krawędziami $\{u_i,v_{2i \bmod 5}\}, i=0,1,2,3,4\}$

$Po_{min}, Po_{sr}, Po_{max}$ - ilość porównań z elementami ciągu w algorytmach sortowania (minimalna, przeciętna i najgorsza)

$Pw_{min}, Pw_{sr}, Pw_{max}$ - ilość przesunięć lub zamian elementów ciągu w danej metodzie sortowania (minimalna, przeciętna, najgorsza)

$\mathcal{Q}$ - zbiór liczb wymiernych,

Q_d - graf d-wymiarowy hipersześcian to graf zawierający 2^d wierzchołków które możemy nazwać kodami Gray'a tak, że kody różniące sie na jednej pozycji są połączone krawędzią

$\mathcal{R}$ - zbiór liczb rzeczywistych

$S(n,k)$ - liczba Stirlinga II rodzaju, jest to ilość podziału zbioru n elementowego na k klas, spełnia rekurencję $S(n+1,k) = S(n,k-1)+kS(n,k)$, dla $1 \leq k < n$, oraz $S(n,1) = S(n,n) = 1, n \geq 1$

$s(n,k)$ - liczba Stirlinga I rodzaju, jest to ilość rozkładów zbioru n elementowego na k cykli, spełnia rekurencję $s(n+1,k) = s(n,k-1)+ns(n,k)$, dla $1 \leq k < n$ oraz $s(n,1) = (n-1)!, s(n,n) = 1, n \geq 1$

SC_n - graf n-ścieżka to graf zawierający n wierzchołków $V(\mathcal{SC}_n) = \{v_1, v_2, ..., v_n\}$ i dokładnie $n-1$ krawędzi $E(\mathcal{SC}_n) = \{\{v_1,v_2\},\{v_2,v_3\},...,\{v_{n-1},v_n\}\}$

$Spec(\alpha)$ - widmo liczby α jest to zbiór $\{\lfloor\alpha\rfloor, \lfloor 2\alpha\rfloor, \lfloor 3\alpha\rfloor, \lfloor 4\alpha\rfloor, ...\}$,

$Spec_i(\alpha)$ - i-ty element widma $Spec(\alpha)$.

$T(L)$ - optymalne drzewo binarne dla ciągu wag L (np. skonstruowane algorytmem Huffmana)

$U(K)$ - rozkład jednostajny na zbiorze K, tzn. dla każdego $x \in K$ jest $P[U(K)=x] = \frac{1}{K}$

$U_{min}, U_{sr}, U_{max}, U^i_{min}, U^i_{sr}, U^i_{max}$ - ilość przesunięć rekordów potrzebnych przy usuwaniu nowego rekordu minimalnie, przeciętnie i w najgorszym przypadku. Litery z indeksem i dotyczą przesuwania nie rekordów ale indeksów

$V(\mathcal{G})$ - zbiór wierzchołków grafu $\mathcal{G}$

V^k_n - ilość wariacji bez powtórzeń k elementów ze zbioru n elementowego, $V^k_n = \frac{n!}{(n-k)!}$

$\overline{V}^k_n$ - ilość wariacji z powtórzeniami k elementów ze zbioru n elementowego, $\overline{V}^k_n = n^k$

$Var(X)$ - wariancja zmiennej losowej X

$W_{min}, W_{sr}, W_{max}, W^i_{min}, W^i_{sr}, W^i_{max}$ - ilość przejrzanych rekordów potrzebnych do znalezienia poszukiwanego rekordu (minimalna, przeciętna i najgorsza). Litery z indeksem i dotyczą przesuwania nie rekordów ale indeksów

W_n - graf n-koło to C_{n-1} cykl z wierzchołkami $V(C_{n-1}) = \{v_1, v_2, ..., v_{n-1}\}$ z dodanym jednym wierzchołkiem v_n który ma krawędzie ł ączące z każdym z z wierzchołków $V(C_{n-1})$

$\mathcal{Z}$ - zbiór liczb całkowitych

$\gamma = 0,57721\,56649\,01533...$ stałą Eulera

$\kappa(\mathcal{G})$ - spójność wierzchołkowa grafu $\mathcal{G}$, jest to najmniejszą liczbę usuniętych wierzchołków powodujących rozspójnienie grafu

$\lambda(\mathcal{G})$ - spójność krawędziowa grafu $\mathcal{G}$, jest to najmniejszą liczbę usuniętych krawędzi powodujących rozspójnienie grafu

τ_n ilość porównań w metodzie sortowania "quicksort"

$\phi(n)$ - funkcja Eulera, oznacza ilość spośród liczb $\{1, 2, ..., n-1\}$ tych liczb, które są względnie pierwsze z n

$\chi_{\mathcal{G}}$ - liczba chromatyczny grafu $\mathcal{G}$ to najmniejsza ilość kolorów potrzeba do pomalowania wierzchołków grafu tak aby sąsiednie wierzchoł ki miały różne kolory

$\overline{\chi}_{\mathcal{G}}$ - indeks chromatyczny grafu $\mathcal{G}$ to najmniejszą ilość kolorów potrzebnych do pomalowania krawędzi tak, aby żadne dwie krawędzie maj ce wspólny wierzchołek nie były tego samego koloru

Υ - macierz incydencji $[\mu_{i,j}]$ grafu $\mathcal{G}$, wierszami są krawędzie a kolumnami wierzchołki, jeśli krawędź wychodzi z danego wierzchołka to piszemy w odpowiedniej kolumnie (-1), jeśli do niego wchodzi piszemy $(+1)$, jeśli wierzchołek nie należy do krawędzi piszemy 0, jeĺi jest to pętla własna piszemy 2

Ω - przestrzeń zdarzeń elementarnych

ω - zdarzenie elementarne

Spis treści

Wstęp **v**

Oznaczenia **vii**

1 Równania rekurencyjne **1**
- 1.1. Zasada indukcji matematycznej . . . 2
- 1.2. Przykłady rekurencji. . . . 5
- 1.3. Metoda repertuaru . . . 11

2 Sumy **17**
- 2.1. Oznaczenia i definicje . . . 18
- 2.2. Metoda czynnika sumacyjnego . . . 20
- 2.3. Własności sumowania . . . 22
- 2.4. Liczby harmoniczne . . . 23
- 2.5. Sumy wielokrotne . . . 25
- 2.6. Różne metody obliczania sum . . . 28
- 2.7. Rachunek różnicowy . . . 32

3 Funkcje całkowitoliczbowe **41**
- 3.1. Podłoga i sufit . . . 42
- 3.2. Widmo liczby . . . 46
- 3.3. Rekurencje z funkcjami podłoga i sufit . . . 47
- 3.4. Działanie dwuargumentowe *mod* . . . 52
- 3.5. Generatory liczb pseudolosowych . . . 54
- 3.6. Zapisy liczb naturalnych w różnych podstawach liczenia . . . 56

4 Teoria liczb **61**
- 4.1. Podzielność, NWD i NWW . . . 62
- 4.2. Liczby pierwsze . . . 65
- 4.3. Reprezentacja liczb wymiernych - drzewo Sterna - Brocota . 70
- 4.4. Kongruencje . . . 73
- 4.5. Funkcja ϕ Eulera i algorytm RSA . . . 76

5 Kombinatoryka **81**
5.1. Permutacje, kombinacje, wariacje 82
5.2. Podziały zbioru i liczby Stirlinga 86
5.3. Współczynniki dwumianowe 90

6 Funkcje tworzące **97**
6.1. Splot 98
6.2. Rozwiązywanie rekurencji za pomocą funkcji tworzących . . . 100
6.3. Algebra wielomianów rzeczywistych 106

7 Wstęp do analizy algorytmów **119**
7.1. Prawdopodobieństwo dyskretne 120
7.2. Asymptotyka 125
7.3. Algorytmy sortowania 129
7.4. Problem rozmieszczenia rekordów w pamięci 137

8 Grafy **153**
8.1. Podstawowe pojęcia i problem reprezentacji 154
8.2. Grafy nieskierowane 158
8.3. Grafy skierowane 177
8.4. Drzewa 179
8.5. Zastosowania teorii grafów 188

Bibliografia **193**

Wykaz tabel **196**

Wykaz rysunków **198**

Wykaz algorytmów **199**

Wykaz osób **201**

Indeks **203**

Rozdział 1

Równania rekurencyjne

1.1. Zasada indukcji matematycznej **2**
1.2. Przykłady rekurencji. **5**
1.3. Metoda repertuaru **11**

1.1. Zasada indukcji matematycznej

Przykład 1.1.1. *Udowodnij, że dla dowolnej liczby naturalnej $n > 0$, liczba $11^n - 3^n$ jest podzielna przez 8.*

Dowód.

Krok 1. $n = 1$ Liczba $11^1 - 3^1 = 8$ jest podzielna przez 8.

Krok 2. Załóżmy, że dla pewnego $n > 0$ oraz pewnej liczby naturalnej k mamy $11^n - 3^n = 8k$. Wtedy $11^{n+1} - 3^{n+1} = 11 \cdot (3^n + 8k) - 3^{n+1} = 88k + (11 - 3) \cdot 3^n = 88k + 8 \cdot 3^n$ jest również liczbą podzielną przez 8.

Stąd wnioskujemy tezę. □

Zasadę indukcji matematycznej możemy więc sformułować następująco

Zasada indukcji matematycznej. *Niech $S(n), n \in \mathcal{N}$, będzie jakimś stwierdzeniem logicznym (to znaczy takim, o którym możemy powiedzieć, że jest prawdziwe lub fałszywe). Jeżeli*

1. *$S(0)$ jest prawdziwe,*

2. *dla każdej liczby naturalnej $k \geq 1$: z prawdziwości $S(k-1)$ wynika prawdziwość zdania $S(k)$,*

to zdanie $S(n)$ jest prawdziwe dla wszystkich liczb naturalnych $\mathcal{N}$.

Krok **1** nazywamy krokiem początkowym a krok **2** krokiem indukcyjnym. Obrazowo działanie indukcji matematycznej można przedstawić na przykładzie ułożonych kostek domina. Jeżeli przewróci się pierwsza kostka (krok początkowy) i wszystkie kostki są tak ustawione blisko siebie, że przewrócenie dowolnej kostki spowoduje przewrócenie następnej, to wszystkie kostki domina się przewrócą. Jednak w tym klasycznym przykładzie nigdy nie jesteśmy w stanie ułożyć wszystkich nieskończenie wielu kostek domino (życia by nie wystarczyło) a jednak dla tego podzbioru zasada indukcji matematycznej działa. Zasada indukcji matematycznej ma wiele różnych wariantów. Na przykład dla naszych kostek mamy:

Zasada indukcji matematycznej (i). *Niech $S(n)$ będzie stwierdzeniem logicznym, takim że*

1. *dla pewnej liczby naturalnej k_o jest prawdziwe zdanie $S(k_o)$,*

2. *dla każdej liczby naturalnej $k > k_o$ i $k < k_1$: z prawdziwości $S(k-1)$ wynika prawdziwość zdania dla $S(k)$.*

Wtedy zdanie $S(n)$ jest prawdziwe dla wszystkich liczb naturalnych $n \in \{k_o, k_o + 1, ..., k_1\}$. □

W terminach zbiorów równoważnym sformułowaniem jest na przykład:

Zasada Indukcji Matematycznej (ii). *Jeżeli $X \subseteq \mathcal{N}$ jest jakimś niepustym podzbiorem liczb naturalnych,*

(i) do którego należy liczba k_o, tzn., $k_o \in X$

(ii) oraz X wraz z każdą liczbą naturalną $k \geq k_o$ zawiera również kolejną liczbę $k+1$, tzn. $\bigwedge_{k \geq k_o} k \in X \Rightarrow k+1 \in X$,
to wtedy zbiór X zawiera wszystkie liczby naturalne $n \geq k_o$, tzn. $X \supseteq \mathcal{N} \backslash \{0, 1, 2, ..., k_o - 1\}$ □

Przykład 1.1.2. *Czasem nie docenia się kroku początkowego. Zwróćmy uwagę, że w ewidentnie fałszywym stwierdzeniu $n = n - 3$ krok indukcyjny jest spełniony bo $k = (k-1)+1 = (k-1)-3+1 = k-3$ a mimo to nie istnieje żadna liczba naturalna dla której to stwierdzenie jest prawdziwe.* □

Problem 1.1.1. *Co by było, gdyby był spełniony krok indukcyjny dla wszystkich naturalnych n i istniała chociaż jedna liczba naturalna dla której takie jak w Przykładzie 1.1.2 stwierdzenie byłoby prawdziwe?* □

Przykład 1.1.3. *Chcemy dowieść twierdzenie, że każda liczba naturalna wi eksza od* 1 *albo jest pierwsza albo można ją przedstawić jako iloczyn liczb pierwszych. Liczba* 2 *jest pierwsza a więc krok początkowy jest spełniony. Jeśli jednak weźmiemy liczbę* 118973 *to informacja, że* 118972 *jest pierwsza lub jest iloczynem liczb pierwszych nic nam nie daje. Musimy mieć silniejsze założenie. Dla dowolnej liczby załóżmy, że nie tylko poprzednią liczbę ale i wszystkie poprzednie można tak przedstawić. Wtedy albo k jest liczbą pierwszą, albo jest złożona, to znaczy, że jest iloczynem dwóch liczb i oraz j takich, że $1 < i, j < k$ a ponieważ z założenia indukcyjnego każda z nich jest albo pierwsza albo jest iloczynem liczb pierwszych, więc i k spełnia tezę.* □

Możemy teraz sformułowć użytą w powyższym przykładzie zasadę:

Zasada indukcji matematycznej zupełnej. *Niech $S(n), n \in \mathcal{N}$, będzie jakimś stwierdzeniem logicznym dla którego*

1. *$S(0)$ jest prawdziwe,*

2. *dla każdej liczby naturalnej $k > 1$: z prawdziwości $S(0), S(1), ..., S(k-1)$ wynika prawdziwość zdania dla $S(k)$.*

Wtedy zdanie $S(n)$ jest prawdziwe dla wszystkich liczb naturalnych $\mathcal{N}$. □

lub równoważne sformułowanie w terminach zbiorów:

Zasada Indukcji Zupełnej. *Jeżeli X jest jakimś niepustym zbiorem liczb naturalnych, który wraz z każdym początkowym fragmentem zbioru $\mathcal{N}$ postaci $\{0, 1, 2, 3, ..., k-1\}$ zawiera również kolejną liczbę k, tzn.* $\bigwedge_{k \in \mathcal{N}}$ *(jeżeli* $\bigwedge_{l<k} l \in X$ *to $k \in X$) to wtedy X zawiera wszystkie liczby naturalne, tzn. $X = \mathcal{N}$.* □

Wydaje się, że to sformułowanie zasady indukcji matematycznej jest ogólniejsze, jednak oba przytoczone tutaj sformułowania są równoważne.

Problem 1.1.2. *Co jest nieprawidłowego w następującym rozumowaniu: Chcemy dowieść że wszystkie samochody kosztują tyle samo. Mamy jakiś jeden samochód i widzimy jego cenę C_1 a więc krok początkowy jest spełniony, Załóżmy, że mamy zbiór k samochodów o cenach $C_1, C_2, ..., C_k$. Ponieważ w zbiorze $C_1, C_2, ..., C_{k-1}$ wszystkie ceny są identyczne a w zbiorze $C_2, C_3, ..., C_k$ też wszystkie ceny są identyczne więc i $C_1, C_2, ..., C_k$ są identyczne, a więc i krok indukcyjny jest prawdziwy. A przecież wystarczy się przejść do salonu samochodowego?* □

Problem 1.1.3. *Czy i jak można sprawdzić, za pomocą indukcji matematycznej, własność która zachodzi dla:*

(a) wszystkich liczb naturalnych podzielnych przez 3?
(b) wszystkich liczb pierwszych?
(c) wszystkich liczb całkowitych?
(d) wszystkich liczb wymiernych?
(e) wszystkich liczb niewymiernych? □

Zasadę indukcji matematycznej używamy najczęściej wtedy, gdy odgadujemy wynik albo gdy znamy zależności pomiędzy k-tym a poprzednimi elementami.

Problem 1.1.4. *Niech będzie dany ciąg taki, że pierwszym jego wyrazem jest $F_1 = 1$, drugim $F_2 = 1$, a każdy następny jest sumą dwóch bezpośrednio poprzednich. Jak można zapisać tę zależność? Czy z podanego stwierdzenia potrafimy wyliczyć każdy wyraz ciągu? Jak to można udowodnić?* □

Problem 1.1.5. *Załóżmy, że mamy algorytm*

Algorytm 1.1. Suma $S(n)$

```
real Function S(n : integer);
  s : real;
  i : integer;
  s:=0;
  for i=1 to n step 1
      s:=s+2 sqrt(s)+1;
  return(s);
```

gdzie funkcja sqrt() oznacza pierwiastek kwadratowy. Ile wyniesie $S(20)$ po wyjściu z tej pętli? Jak można wypisać zależność s_n od s_{n-1} i jak to równanie rekurencyjne można rozwiązać? □

1.2. Przykłady rekurencji.

W Problemie 1.1.5 podaliśmy algorytm obliczania pewnego ciągu. Porównajmy ten algorytm z następującymi dwoma algorytmami:

Algorytm 1.2. Suma $S(n)$ - drugi sposób

```
real Function S(n : integer);
  if n=0 then
     return (0)
  else
     return(S(n-1)+2sqrt(S(n-1))+1);
```

oraz

Algorytm 1.3. Suma $S(n)$ - trzeci sposób

```
real Function S(n : integer);
   return n*n;
```

Rozważaną we wszystkich trzech algorytmach rekurencją jest rekurencja

$$\begin{cases} s_0 &= 1, \\ s_n &= s_{n-1} + 2\sqrt{s_{n-1}} + 1 \ \text{ dla } \ n \geq 1, \end{cases}$$

natomiast jej rozwiązaniem $s_n = n^2, n \geq 0$, (sprawdź indukcyjnie).

Widzimy więc, że równania rekurencyjne związane są z pętlami w językach programowania i można je zawsze implementować za pomocą pewnego zestawu pętli (zob. Algorytm 1.1).

Możemy również równanie rekurencyjne zaimplementować (zob. Algorytm 1.2) jako funkcję rekurencyjną (to znaczy taką, która sama siebie wywołuje). W przypadku procedur/funkcji rekurencyjnych istnieje szereg pułapek, na które doświadczony programista powinien zwracać uwagę. Tutaj, na przykład, do wyliczenia $S(n)$ potrzebne jest dwukrotne obliczenie tej samej wartości $S(n-1)$, a więc dwa razy wywołuje się obliczającą to samo procedurę. Przy dalszych wywołaniach błąd ten się bardzo szybko kumuluje. Błąd w tym algorytmie jest oczywisty, jednak są algorytmy, gdzie dubluje się obliczanie tych samych wartości, ale zweryfikowanie tego jest trudne. Rekurencja w programowaniu może też dramatycznie zwiększyć złożoność obliczeniową. Ponadto rekurencja zawsze zwiększa pamięciowe zapotrzebowanie programu (chyba że zostanie użyta możliwa w pewnych przypadkach optymalizacja zwana rekursją ogonową), gdyż wymaga ona zapamiętania m.in. adresów powrotu, pozwalających programowi "zorientować się" do którego miejsca ma wrócić po zakończeniu jednego z wywołań rekurencyj-

nych. Niezaprzeczalną zaletą rekurencji jest przejrzystość programów, które z niej korzystają.

W Algorytmie 1.3 wykorzystano rozwiązanie tego równania rekurencyjnego.

Oczywiście najoptymalniejszy pod każdym względem jest Algorytm 1.3.

Równanie rekurencyjne jest to równanie, w którym znana jest pewna ilość początkowych wyrazów ciągu $(s_o, s_1, s_2, ..., s_k)$, a ogólny wyraz s_n wyliczany jest w oparciu o znajomość kilku lub wszystkich poprzednich wyrazów $(s_o, s_1, s_2, ..., s_{n-1})$.

Rozwiązanie równania rekurencyjnego jest to zapisanie wyrazu s_n zwartym wzorem zależnym tylko od argumentu n. Rozwiązanie równania rekurencyjnego zazwyczaj znacznie skraca i upraszcza obliczanie dowolnego n -tego elementu.

Przykład 1.2.1. Silnia $n!$ *Definiujemy* $0! = 1, n! = 1 \cdot 2 \cdot 3 \cdot ... \cdot n, n \geq 1$. *Aby jednak komputer był w stanie efektywnie wyliczyć silnię, potrzebna jest inna definicja:*

$$\begin{cases} s_0 &= 1, \\ s_n &= ns_{n-1} \ \ dla \ \ n \geq 1, \end{cases}$$

wtedy $s_n = n!, n \geq 0$. *Czy ta defincja jest poprawna (por. z zasadą indukcji matematycznej)? Co będzie, jak w tej definicji zmienimy* $s_o = 1$ *na* $s_o = 0$ *lub* $s_o = \frac{1}{2}$*?* □

Przykład 1.2.2. Ciągi arytmetyczne i geometryczne. *Ciąg nazywamy arytmetycznym jeżeli różnica każdego wyrazu i wyrazu poprzedniego ciągu jest stała, równa* r*:*

$$\begin{cases} a_0 &= \alpha, \\ a_n &= a_{n-1} + r = a_0 + nr \ \ dla \ \ n \geq 1. \end{cases}$$

Sumę ciągu arytmetycznego liczymy ze wzoru:

$$s_n = s_{n-1} + a_n = s_{n-1} + a_0 + nr = (n+1)\alpha + \frac{n(n+1)}{2}r \ \ dla \ \ n \geq 0.$$

Analogicznie ciąg nazywamy geometrycznym, jeżeli iloraz każdego wyrazu przez wyraz poprzedni jest stały równy q*:*

$$\begin{cases} a_0 &= \alpha, \\ a_n &= qa_{n-1} = \alpha q^n \ \ dla \ \ n \geq 1. \end{cases}$$

Sumę ciągu geometrycznego liczymy ze wzoru

$$s_n = s_{n-1} + a_n = \alpha \frac{1 - q^{n+1}}{1 - q} \ \ dla \ \ n \geq 0.$$

□

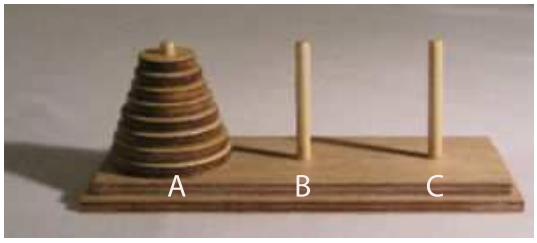
Przykład 1.2.3. "Problem cyfr". *Jak wiele jest łańcuchów n-cyfrowych zawierających parzystą liczbę 0? Niech a_n będzie ilością takich łańcuchów n-cyfrowych. Aby otrzymać z łańcucha $n-1$ cyfrowego łańcuch n cyfrowy należy dodać "0" do któregoś z $10^{n-1} - a_{n-1}$ łańcuchów nie zawierających parzystej liczby 0 albo dodać jedną z 9 cyfr do któregoś z a_{n-1} ciągów. Stąd mamy*

$$\begin{cases} a_1 &= 9, \\ a_n &= (10^{n-1} - a_{n-1}) + 9a_{n-1} = 10^{n-1} + 8a_{n-1} \ \ dla \ \ n \geq 1, \end{cases} \tag{1.2.1}$$

a stąd $a_4 = 7048$. Jakie jest ogólne rozwiązanie tej rekurencji? (zob. Tabela 1.7) □

Przykład 1.2.4. Wieża w Hanoi. (E. Lucas, 1883)

Rysunek 1.1. Wieża w Hanoi.



U zarania czasu Bóg umieścił 64 złote krążki na jednej z trzech diamentowych iglic tak, że krążki najniżej położone miały największe promienie a te najwyżej położone najmniejsze promienie (zob. Rys. 1.1). Następnie Bóg polecił grupie mnichów przełożenie tych krążków na drugą iglicę (B), ale tak by:

- *w jednym ruchu przenosić tylko jeden krążek,*
- *krążek większy nigdy nie może leżeć na krążku mniejszym,*
- *można posługiwać się trzecią iglicą C.*

Mnisi pracują od zarania dziejów dzień i noc ... . Przełożenie jednego krążka zajmuje im 1 sek. Jak można to zrobić i ile czasu im to zajmie? (Według legendy nastąpi wtedy koniec świata). Niech `HANOI`(A, B, n) *oznacza zbiór ruchów jakie należy wykonać przenosząc n krążków z A do B gdzie C może być swobodnie wykorzystywany, C może być pusty albo zawierać prążki większe od wszystkich przenoszonych prążków. Przeniesienie krążka z A do B oznaczać będziemy przez $A \to B$.*

Dla **n=0** *nie trzeba nic robić:*

`HANOI(A,B,0)`$=\emptyset$

Dla **n=1** *mamy jeden ruch $A \to B$:*

`HANOI(A,B,1)`$=\{A \to B\}$

Dla **n=2** *mamy trzy ruchy* $A \to C, A \to B, C \to B$:
`HANOI(A,B,2)=HANOI(A,C,1)∪HANOI(A,B,1)∪HANOI(C,B,1)=`$\{A \to C, A \to B, C \to B\}$
Dla **n=3** *potrzeba 7 ruchów: najpierw przenosimy dwa górne krążki na C* $(A \to B, A \to C, B \to C)$ *następnie największy krążek z A przenosimy na B* $(A \to B)$ *a następnie 2 krążki z C przenosimy na B* $(C \to A, C \to B, A \to B)$*:*
`HANOI(A,B,3)=HANOI(A,C,2)∪HANOI(A,B,1)∪HANOI(C,B,2)=`$\{A \to B, A \to C, B \to C, A \to B, C \to A, C \to B, A \to B\}$
Dla **n=4** *potrzeba 15 ruchów:*
`HANOI(A,B,4)=HANOI(A,C,3)∪HANOI(A,B,1)∪HANOI(C,B,3)=`$\{A \to C, A \to B, C \to B, A \to C, B \to A, B \to C, A \to CA \to B, C \to B, C \to A, B \to A, C \to B, A \to C, A \to B, C \to B\}$
Z tych przykładów widzimy jak można to zrobić:

$$\begin{cases} \texttt{HANOI}(A,B,0) &= \emptyset, \\ \texttt{HANOI}(A,B,n) &= \texttt{HANOI}(A,C,n-1) \cup \texttt{HANOI}(A,B,1) \\ & \quad \cup \texttt{HANOI}(C,B,n-1), \quad dla \;\; n \geq 1, \end{cases}$$

a stąd liczba niezb ednych przełożeń h_n *spełnia:*

$$\begin{cases} h_o &= 0, \\ h_n &= h_{n-1} + 1 + h_{n-1} = 2h_{n-1} + 1, \quad dla \;\; n \geq 1. \end{cases}$$

Obliczając kilka pierwszych elementów tak określonego ciągu otrzymujemy:

Tabela 1.1. Ilość ruchów potrzebnych na przeniesienie prążków w problemie wieży z Hanoi

n	0	1	2	3	4	5	6	7	8	9	10	11	12
h_n	0	1	3	7	15	31	63	127	255	511	1023	2047	4095

Żeby rozwiązać to równanie rekurencyjne dodajmy do obu stron drugiego równania 1 *i zróbmy podstawienie* $u_n = h_n + 1, n \geq 0$, *wtedy*

$$\begin{cases} u_o &= h_o + 1 = 1, \\ u_n = h_n + 1 &= 2h_{n-1} + 2 = 2u_{n-1}, \quad dla \;\; n \geq 1, \end{cases}$$

czyli

$$\begin{cases} u_o &= 1, \\ u_n &= 2u_{n-1}, \quad dla \;\; n \geq 1, \end{cases}$$

a to jest równanie rekurencyjne ciągu geometrycznego (porównaj z Przykładem 1.2.2), czyli $u_n = 2^n$ *a stąd* $h_n = 2^n - 1, n \geq 0$. □

Problem 1.2.1. Problemy związane z wieżą z Hanoi. *Jak postępować, jeśli mamy do dyspozycji cztery a nie trzy pręty? Jak wygląda rekurencja gdy mamy $2n$ krążków z których każde dwa są tych samych rozmiarów?* □

Przykład 1.2.5. Problem pizzy. (Proste na płaszczyźnie.)

Jaka jest największa możliwa liczba l_n obszarów wyznaczonych przez n prostych na płaszczyźnie? (Jak najmniejszą liczbą ruchów podzielić pizzę na l_n kawałków?)

Sprawdźmy najpierw kilka początkowych wartości.

(i) Gdy nie ma żadnej prostej obszar jest jeden $l_o = 1$.

(ii) Jedna prosta tworzy zawsze dwa różne obszary $l_1 = 2$.

(iii) Kładąc drugą prostą (byle nie równoległą do pierwszej) otrzymujemy $l_2 = 4$ obszary.

Jednak wcale nie jest $l_n = 2^n$ bo $l_3 = 7$!

Tabela 1.2. Ilość obszarów które można uzyskać w problemie pizzy.

n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
l_n	1	2	4	7	11	16	22	29	37	46	56	67	79	92	106

Zauważmy, że nowa prosta zwiększa ilość obszarów o k jeśli przecina dokładnie $k-1$ poprzednich prostych i to w nowych punktach przecięć. Z drugiej strony dwie proste mogą się przeciąć co najwyżej w jednym punkcie i przecinają się o ile nie są równoległe. Widzimy zatem, że najwięcej obszarów dostaniemy kładąc kolejne proste w ten sposób, aby żadne dwie nie były równoległe i żadne trzy nie przecinały się w jednym punkcie. Otrzymujemy następujące równanie rekurencyjne:

$$\begin{cases} l_o &= 1, \\ l_n &= l_{n-1} + n, \ \ dla \ n \geq 1, \end{cases}$$

Porównując ten wzór ze wzorem na sumę ciągu arytmetycznego (Przykład 1.2.2 $a_o = 0, \alpha = s_o = 1, r = 1$) otrzymujemy $l_n = 1 + \frac{n(n+1)}{2}, n \geq 0$. □

Problem 1.2.2. Inne problemy związane z problemem pizzy: *Wyznacz największą liczbę kawałków ograniczonych n konfiguracjami (tniemy nożem o kształcie):*

(a) konfiguracja kątów (w kształcie litery V),

(b) konfiguracja dwóch okręgów stycznych wewnętrznie,

(c) konfiguracja dwóch okręgów stycznych zewnętrznie,

(d) konfiguracja trzech okręgów stycznych wewnętrznie,

(e) konfiguracja dwóch prostych równoległych,

(f) konfiguracja trzech prostych równoległych. □

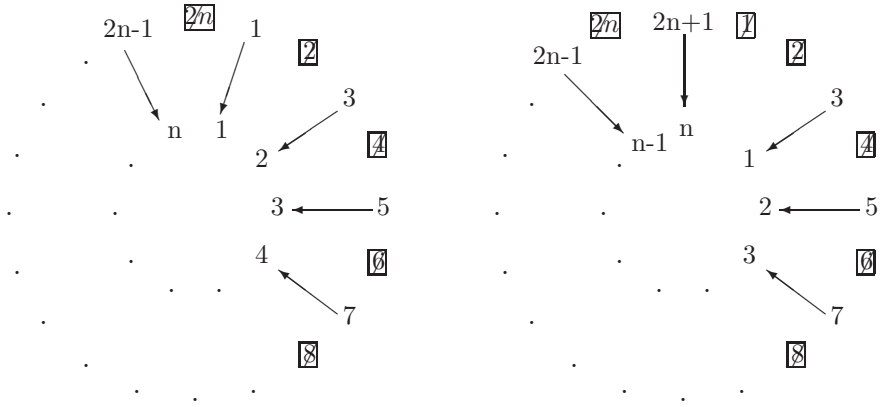
Przykład 1.2.6. Problem Flawiusza. *41 żydowskich powstańców zamkniętych jest przez Rzymian w jaskini. Aby nie dostać się w ręce wrogów powstańcy postanowili sami się zabić. Ustawili się w okrąg z wyróżnioną jedną osobą i zaczęli liczyć od tej wyróżnionej osoby co 3 osobę, zabijając ją. Na której pozycji należy się ustawić, aby zostać ostatnią osobą i uniknąć śmierci? Ogólnie mamy n osób i eliminujemy co k - tą osobę. Na której pozycji $J_k(n)$ należy się ustawić, aby pozostać? Będziemy w skrócie pisać $J(n) = J_2(n)$ i zajmiemy się w tym rozdziale tylko rekurencją $J(n)$ (inne przypadki zostaną omówione w rodziale 3.2).*

Rozważając początkowe przypadki otrzymujemy:

Tabela 1.3. Numer pozycji na której zostanie żywa osoba w problemie Flawiusza

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
J(n)	1	1	3	1	3	5	7	1	3	5	7	9	11	13	15

Rysunek 1.2. Eliminacje po pierwszym obiegu okręgu w Problemie Flawiusza.



Rozważając przenumerowanie po pierwszym obiegu okręgu (Rys. 1.2) otrzymujemy rekurencję:

$$\begin{cases} J(1) & = & 1, \\ J(2n) & = & 2J(n) - 1, \quad dla \ n \geq 1, \\ J(2n+1) & = & 2J(n) + 1, \quad dla \ n \geq 1. \end{cases}$$

□

Niech n będzie dowolną liczbą naturalną a m, l będą liczbami określonymi nierównościami $0 \leq l < 2^m, 2^m \leq n < 2^{m+1}$. Liczby te, dla ustalonego n są zadane jednoznacznie ($m = \lfloor \log_2 n \rfloor, l = n - 2^m$). W tym przykładzie

wszędzie przez $m = m(n)$ i $l = l(n)$ będziemy rozumieć tak otrzymane z n liczby.

Twierdzenie 1.2.1. *Zachodzi* $J(n) = J(2^m + l) = 2l + 1, n \geq 1$.

Dowód. (Indukcja względem m)

Krok 1. $m = 0$.

Wtedy $n = 2^0 + l$ a $l < 2^0$ czyli $l = 0$ oraz $n = 1$ a $J(1) = 1 = 2 \cdot 0 + 1$.

Krok 2. Zakładamy prawdziwość twierdzenia dla pewnego m i dowodzimy prawdziwości dla $m + 1$.

Rozważmy dwa przypadki:

Przypadek 1. l jest liczbą parzystą. Wtedy

$$J(2^{m+1} + l) = 2J(2^m + \frac{l}{2}) - 1 = 2(2 \cdot \frac{l}{2} + 1) - 1 = 2l + 1.$$

Przypadek 2. l jest liczbą nieparzystą.

$$J(2^{m+1} + l) = 2J(2^m + \frac{l-1}{2}) + 1 = 2(2 \cdot \frac{l-1}{2} + 1) + 1 = 2l + 1.$$

a więc teza zachodzi też dla $m + 1$ i dowolnego $0 \leq l < 2^{m+1}$. □

Uwaga 1.2.2. System binarny. *Niech n będzie przedstawione w układzie binarnym (dwójkowym) $n = (b_m b_{m-1} ... b_2 b_1 b_o)_2$, gdzie najbardziej znaczący bit $b_m = 1$. Wtedy $l = (0 b_{m-1} ... b_2 b_1 b_0)_2$ oraz*

$$J(n) = 2l + 1 = (b_{m-1} b_{m-2} ... b_2 b_1 b_0 1)_2 = (b_{m-1} b_{m-2} ... b_2 b_1 b_0 b_m)_2.$$

Jest to więc cykliczne przesunięcie reprezentacji binarnej. □

1.3. Metoda repertuaru

Uogólnijmy rekurencję Flawiusza (skomplikujmy problem) rozważając klasę równań rekurencyjnych zależnych od parametrów $\alpha, \beta, \gamma \in \mathcal{R}$.

$$\begin{cases} f(1) & = & \alpha, \\ f(2n) & = & 2f(n) + \beta, \ \text{dla} \ n \geq 1, \\ f(2n+1) & = & 2f(n) + \gamma, \ \text{dla} \ n \geq 1. \end{cases}$$

Oczywiście do zwykłej rekurencji Flawiusza wracamy przyjmując $\alpha = \gamma = 1, \beta = -1$. Obliczmy kilka początkowych wyrazów:

Tabela 1.4. Kolejne rozwiązania uogólnionego problemu Flawiusza

n	$f(n)$	n	$f(n)$	n	$f(n)$
1	α	12	$8\alpha + 3\beta + 4\gamma$	23	$16\alpha + 8\beta + 7\gamma$
2	$2\alpha + \beta$	13	$8\alpha + 2\beta + 5\gamma$	24	$16\alpha + 7\beta + 8\gamma$
3	$2\alpha + \gamma$	14	$8\alpha + \beta + 6\gamma$	25	$16\alpha + 6\beta + 9\gamma$
4	$4\alpha + 3\beta$	15	$8\alpha + 7\gamma$	26	$16\alpha + 5\beta + 10\gamma$
5	$4\alpha + 2\beta + \gamma$	16	$16\alpha + 15\beta$	27	$16\alpha + 4\beta + 11\gamma$
6	$4\alpha + \beta + 2\gamma$	17	$16\alpha + 14\beta + \gamma$	28	$16\alpha + 3\beta + 12\gamma$
7	$4\alpha + 3\gamma$	18	$16\alpha + 13\beta + 2\gamma$	29	$16\alpha + 2\beta + 13\gamma$
8	$8\alpha + 7\beta$	19	$16\alpha + 12\beta + 3\gamma$	30	$16\alpha + \beta + 14\gamma$
9	$8\alpha + 6\beta + \gamma$	20	$16\alpha + 11\beta + 4\gamma$	31	$16\alpha + 15\gamma$
10	$8\alpha + 5\beta + 2\gamma$	21	$16\alpha + 10\beta + 5\gamma$		
11	$8\alpha + 4\beta + 3\gamma$	22	$16\alpha + 9\beta + 6\gamma$	$\vdots$	$\vdots$

Zauważmy, że

$$f(n) = A(n) \cdot \alpha + B(n) \cdot \beta + C(n) \cdot \gamma, \quad n \geq 1.$$

Ogólna strategia przy rozwiązywaniu równań rekurencyjnych metodą repertuaru może być opisana następująco:

- Uogólnij równanie rekurencyjne uzależniając je od parametrów
- Ustal parametry dla których znasz rozwiązania
- Połącz przypadki szczególne aby znaleźć rozwiązanie ogólne.

⋆ **Pierwszy przypadek szczególny** $f(n) = 1$**.** Wtedy

$$\begin{cases} f(1) & = & 1 = \alpha, \\ f(2n) & = & 1 = 2 + \beta, \ \text{dla} \ n \geq 1, \\ f(2n+1) & = & 1 = 2 + \gamma, \ \text{dla} \ n \geq 1, \end{cases}$$

czyli $\alpha = 1, \beta = \gamma = -1$ - można znaleźć parametry stanowiące rozwiązanie tego równania niezależne od n. Jest to więc dobre rozwiązanie. Stąd otrzymujemy:

$$1 = A(n) - B(n) - C(n), n \geq 1. \tag{1.3.2}$$

⋆ **Drugi przypadek szczególny** $f(n) = n$. Wtedy

$$\begin{cases} 1 & = & \alpha, \\ 2n & = & 2n + \beta, \ \text{dla} \ n \geq 1, \\ 2n+1 & = & 2n + \gamma, \ \text{dla} \ n \geq 1, \end{cases}$$

a stąd $\alpha = \gamma = 1, \beta = 0$ co daje:

$$n = A(n) + C(n), n \geq 1. \tag{1.3.3}$$

⋆ **Przypadek szczególny (zły)** $f(n) = n^2$. Wtedy

$$\begin{cases} 1 & = & \alpha, \\ 4n^2 & = & 2n^2 + \beta, \ \text{dla} \ n \geq 1, \\ 4n^2 + 4n + 1 & = & 2n^2 + \gamma, \ \text{dla} \ n \geq 1, \end{cases}$$

i nie istnieją parametry β i γ niezależne od n i takie, aby powyższe równanie zachodziło dla wszystkich n.

⋆ **Trzeci przypadek szczególny** $f(n) = 2^{m(n)}$**.** Wtedy uwzględniając nierówności:

$$\begin{aligned} &2^0 \leq 1 < 2, \\ &2^{m(n)+1} \leq 2n < 2n + 1 < 2^{m(n)+2}, \end{aligned}$$

otrzymujemy

$$\begin{cases} 2^0 & = & \alpha, \\ 2^{m+1} & = & 2^{m+1} + \beta, \ \text{dla} \ n \geq 1, \\ 2^{m+1} & = & 2^{m+1} + \gamma, \ \text{dla} \ n \geq 1, \end{cases}$$

a stąd $\alpha = 1, \beta = \gamma = 0$, a więc

$$A(n) = 2^m, \quad \text{gdzie} \quad n = 2^m + l, n \geq 1. \tag{1.3.4}$$

Ze wzorów (1.3.2)-(1.3.4) otrzymujemy:

$$\begin{cases} A(n) & = & 2^m, \\ 1 & = & A(n) - B(n) - C(n), \\ n & = & A(n) + C(n), \quad n \geq 1, \end{cases}$$

a stąd $A(n) = 2^m, B(n) = 2^m - l - 1, C(n) = n - 2^m = l$, a więc

$$f(n) = \alpha 2^m + \beta(2^m - l - 1) + \gamma l$$

i wstawiając $\alpha = \gamma = 1, \beta = -1$, otrzymujemy

$$J(n) = 2^m - 2^m + l + 1 + l = 2l + 1. \qquad \square$$

Widać było w tym przykładzie, że o ile funkcje $f(n) = 1$ czy $f(n) = n$ dość łatwo odgadnąć, o tyle funkcja $f(n) = 2^m$ jest bardzo specyficzna i trudno zgadnąć, że ją należy wstawić.

Podamy teraz inny przykład zastosowania metody repertuaru.

Przykład 1.3.1. *Rozwiążmy równanie rekurencyjne:*

$$\begin{cases} T(1) &= 3, \\ T(n) &= 3T(n-1)+n-2, \quad n \geq 2. \end{cases}$$

Uogólniamy to równanie do

$$\begin{cases} f(1) &= \alpha, \\ f(n) &= 3f(n-1)+\beta n+\gamma, \quad n \geq 2, \end{cases}$$

i rozważając przypadki szczególne otrzymujemy:

Tabela 1.5. Wyniki metody repertuaru dla rekurencji.

$f(n)=1$	$f(n)=n$	$f(n)=3^n$
$\begin{cases} 1 &= \alpha, \\ 1 &= 3+\beta n+\gamma, \end{cases}$	$\begin{cases} 1 &= \alpha, \\ n &= 3n-3+\beta n+\gamma, \end{cases}$	$\begin{cases} 3 &= \alpha, \\ 3^n &= 3^n+\beta n+\gamma, \end{cases}$
$\begin{cases} \alpha &= 1, \\ \beta &= 0, \\ \gamma &= -2, \end{cases}$	$\begin{cases} \alpha &= 1, \\ \beta &= -2, \\ \gamma &= 3, \end{cases}$	$\begin{cases} \alpha &= 3, \\ \beta &= 0, \\ \gamma &= 0, \end{cases}$
$1 = A(n)-2C(n),$	$n = A(n)-2B(n)+3C(n),$	$3^n = 3A(n),$

skąd

$$\begin{cases} A(n) &= 3^{n-1}, \\ B(n) &= \frac{5}{4}3^{n-1}-\frac{3}{4}-\frac{n}{2}, \\ C(n) &= \frac{3^{n-1}-1}{2}, \quad n \geq 1, \end{cases}$$

co prowadzi do

$$f(n) = \alpha 3^{n-1}+\beta(\frac{5}{4}3^{n-1}-\frac{3}{4}-\frac{n}{2})+\gamma\frac{3^{n-1}-1}{2}, \quad n \geq 1,$$

czyli

$$T(n) = 3^n+\frac{1}{4}3^{n-1}+\frac{1}{4}-\frac{n}{2}, \quad n \geq 1. \qquad \square$$

Przykład 1.3.2. *Metoda ta, mimo najszczerszych chęci, może nie prowadzić do rozwiązania. Rozważmy Przykład 1.2.3. Uogólniamy równanie rekurencyjne (1.2.1) do*

$$\begin{cases} a(1) &= \alpha, \\ a(n) &= \beta a(n-1)+\gamma 10^{n-1}, \quad n \geq 2. \end{cases} \tag{1.3.5}$$

i rozważając przypadki szczególne otrzymujemy:

Tabela 1.6. Zastosowanie metody repertuaru do problemu cyfr.

$a(n)=1$	$a(n)=2^n$	$a(n)=10^n$
$\begin{cases} 1 = \alpha, \\ 1 = \beta+\gamma 10^{n-1}, \end{cases}$	$\begin{cases} 2^1 = \alpha, \\ 2^n = \beta 2^{n-1}+\gamma 10^{n-1}, \end{cases}$	$\begin{cases} 10^1 = \alpha, \\ 10^n = \beta 10^{n-1}+\gamma 10^{n-1}, \end{cases}$
$\begin{cases} \alpha = 1, \\ \beta = 1, \\ \gamma = 0, \end{cases}$	$\begin{cases} \alpha = 2, \\ \beta = 2, \\ \gamma = 0, \end{cases}$	$\begin{cases} \alpha = 10, \\ \beta = 0, \\ \gamma = 10, \end{cases}$
$1 = A(n)+B(n),$	$2^n = 2A(n)+2B(n),$	$10^n = 10A(n)+10C(n),$

otrzymaliśmy więc sprzeczny układ równań. Przenalizujmy dlaczego się tak stało. Rozważając kolejne rozwiązania sparametryzowanego układu otrzymujemy:

Tabela 1.7. Rozwiązania uogólnionego problemu cyfr

1	γ
2	$10\alpha+\beta\gamma$
3	$10^2\alpha+10\alpha\beta+\beta^2\gamma$
4	$10^3\alpha+10^2\alpha\beta+10\alpha\beta^2+\beta^3\gamma$
$\vdots$	$\vdots$
k	$\alpha\sum_{j=1}^{k-1}10^{k-j}\beta^{j-1}+\beta^{k-1}\gamma$
$\vdots$	$\vdots$

a jak widać rozwiązania te nie są liniowymi kombinacjami α,β,γ. □

Uwaga 1.3.1. *Skąd się bierze taka metoda postępowania z równaniami rekurencyjnymi? W zasadzie ideą, która leży u podstaw metody repertuaru jest zgadywanie. Jeśli jednak mamy konkretne równanie rekurencyjne $t_n = f_n(t_1,...,t_{n-1}), n \geq 0$, to odgadnąć, czy rozwiązaniem jest $t_n = 3.257n^2 - 0.45$, czy $t_n = 3.251n^2 - 0,44$ czy też z innymi współczynnikami liczbowymi jest prawie niemożliwe. Jednak mamy inną sytaucję, gdy zastanawiamy się nie nad konkretnym rozwiązaniem, ale nad tym, z jakiej klasy funkcje "wchodzą" w skład rozwiązania? Dokładnie, chcemy rozważać nie pojedyncze równanie rekurencyjne ale klasę równań rekurencyjnych $\mathcal{K}(E) = \{\{T_n = f_{n,\alpha,\beta,\gamma...}(T_1,...,T_{n-1})\} : \alpha,\beta,\gamma,... \in E\}$, taką, że $\{t_n, n \geq 1\} \in \mathcal{K}(E)$. Jeżeli klasa $\mathcal{K}(E)$ jest dostatecznie szeroka, to być może należą do niej pewne klasy funkcji: $1, n, n^2, n^3, n^4, ..., 2^n, 3^n, ..., n!$, stanowiące "bazę" interesującego*

nas rozwiązania, a wtedy konkretne rozwiązanie $\{t_n, n \geq 1\}$ *jest liniową kombinacją tych, sprawdzonych przez nas funkcji "bazowych".* □

Uwaga 1.3.2.

(i) Parametryzując równanie rekurencyjne, które zawiera funkcje np. $1, n^2$, *czasem lepiej jest dorzucić jeszcze jeden parametr z funkcją* n. *Na przykład* $T_n = T_{n-1} + 3n^2 - 5, n \geq 1$, *dobrze jest sparametryzować do* $T_n = T_{n-1} + \alpha n^2 + \beta n + \gamma, n \geq 1$. *Podobnie, jeżeli występuje wyraz* n *w najwyższej potędze* k *to niezależnie od innych wyrazów, przy parametryzacji należy rozważać wielomian* $\alpha n^k + \beta n^{k-1} + \gamma n^{k-2} + \ldots$.

(ii) Jeżeli w równaniu rekurencyjnym przy T_{n-1} *występuje zależny od* n *wyraz* a_n *to należy również rozważyć funkcję* $\prod_{k=1}^{n} a_k$. *W szczególności dla rekurencji* $T_n = nT_{n-1} + 2n^4 - 4, n \geq 1$, *należy rozważyć funkcje:* $1, n, n^2, n^3, n^4, n!$, *a w rekurencji* $T_n = 3T_{n-1} + 2n, n \geq 1$, *funkcje* $1, n, 3^n$. □

Podsumowanie

- **Definicje rekurencyjne**. Definicja rekurencyjna (indukcyjna):
  nieformalnie - taka definicja, która odwołuje się do samej siebie - ale trzeba tu uważać, by odwołanie było do instancji o mniejszej komplikacji,
  zwykle chodzi o ciąg - dla którego przepis na element wykorzystuje jakieś poprzednie elementy,
  początkowy element (lub kilka początkowych) muszą być zadane konkretnie - żeby było od czego zacząć,
  zwykle definicja rekurencyjna odwołuje się do jednego lub kilku poprzednich elementów, ale może też odwoływać się do wszystkich poprzednich.
- Poprawność definicji rekurencyjnych wynika z zasady indukcji matematycznej
- Rozwiązanie rekurencji zazwyczaj, choć nie zawsze, przyspiesza obliczenia programów komputerowych
- Poprzez sparametryzowanie stałych występujących w równaniu rekurencyjnym i sprawdzanie jakie klasy funkcji wchodzą w skład sparametryzowanego równania możemy rozwiązać równanie rekurencyjne metodą repertuaru.

Rozdział 2

Sumy

2.1. Oznaczenia i definicje **18**
2.2. Metoda czynnika sumacyjnego **20**
2.3. Własności sumowania **22**
2.4. Liczby harmoniczne **23**
2.5. Sumy wielokrotne **25**
2.6. Różne metody obliczania sum **28**
2.7. Rachunek różnicowy **32**

2.1. Oznaczenia i definicje

Symbol reprezentujący sumy, używający greckiej dużej litery $\sum$ (sigma), był wprowadzony przez matematyka Josepha Fouriera w 1820 roku.

Definicja 2.1.1. *Wyrażenie* $\sum_{k=m}^{n} a(k), m, n \in \mathcal{Z}$, *gdzie* $a(.)$ *jest funkcją określoną na* $[m,n] \cap \mathcal{Z}$, *przyjmuje wartość* $a(m) + a(m+1) + \ldots + a(n-1) + a(n)$, *jeżeli* $m \leq n, m, n, \in \mathcal{Z}$ *i* 0 *w pozostałych przypadkach.* □

Funkcja $a(.)$ określona na podzbiorze liczb całkowitych jest nazywana zazwyczaj ciągiem i stosuje się skrócony zapis $a_k = a(k), k \in \mathcal{Z}$.

Często zamiast zapisu $\sum_{k=m}^{n} a_k$, stosuje się zapis $\sum_{\{k \in \mathcal{Z}: m \leq k \leq n\}} a_k$, lub krócej $\sum_{m \leq k \leq n} a_k$, wiedząc, że zbiór znajdujący się pod znakiem sumy jest zawsze podzbiorem $\mathcal{Z}$. Należy uważać jednak, bo ten ostatni zapis sumy nie jest jednoznaczny, nie wiemy czy indeksem sumowania jest k, n czy m! Na przykład:

$$\sum_{k<m,k,m \in \mathcal{N}} \frac{k}{2^m} = \begin{cases} \frac{k}{2^k}, & \text{jeśli indeksem sumowania jest } m, \\ \frac{(m-1)m}{2^{m+1}}, & \text{jeśli indeksem sumowania jest } k. \end{cases}$$

Stąd mamy różne postacie tej samej sumy:

$$a_1 + a_2 + a_3 + ... + a_n, \quad \sum_{k=m}^{n} a_k, \quad \sum_{m \leq k \leq n} a_k, \quad \sum a_k [m \leq k \leq n].$$

Wyjaśnimy teraz ten ostatni zapis. Niech $\mathcal{P}(k)$ będzie pewną funkcją zdaniową, prawdziwą lub fałszywą, dla dowolonej liczby całkowitej k, wtedy piszemy

$$\sum_{\{k \in \mathcal{Z}: \mathcal{P}(k) \text{ jest prawdziwe}\}} a_k = \sum_{\mathcal{P}(k)} a_k = \sum a_k [\mathcal{P}(k)],$$

gdzie

$$[\mathcal{P}(k)] = \begin{cases} 1, & \mathcal{P}(k) \text{ jest prawdziwe}, \\ 0, & \mathcal{P}(k) \text{ jest fałszywe}, \end{cases}$$

przyjmując jednak $0 \cdot \infty = 0$. Zapis ten nazywa się zapisem sumy wg. konwencji K. E. Iversona.

Przykład 2.1.1. *Niech*

$$[k \textit{ jest liczbą pierwszą}] = \begin{cases} 1, & k \textit{ jest liczbą pierwszą}, \\ 0, & k \textit{ nie jest liczbą pierwszą}. \end{cases}$$

Wtedy

$$\sum_k [k \textit{ jest liczbą pierwszą}][1 \leq k \leq n] \frac{1}{k},$$

oznacza sumę odwrotności wszystkich liczb pierwszych spośród $\{1, 2, 3, 4, ..., n\}$. *Warto zwrócić uwagę, że chociaż odwrotność* 0 *nie istnieje, to ponieważ* $[0 \text{ jest liczbą pierwszą}] = [1 \leq 0 \leq n] = 0$ *więc umownie przyjmuje się*

$$[0 \text{ jest liczbą pierwszą}][1 \leq 0 \leq n]\frac{1}{0} = 0 \cdot \infty = 0. \qquad \square$$

Różne postacie sum mogą być wygodne i poręczne przy wielu różnych przekształceniach.

Związek pomiędzy sumami a rekurencjami. *Każda suma jest rekurencją. Rzeczywiście*

$$S_n = \sum_{k=0}^{n} a_k, \quad \begin{cases} S_o &= a_o, \\ S_n &= S_{n-1} + a_n, n \geq 1, \end{cases}$$

Wiele rekurencji daje się zapisać w postaci sum, np. rekurencja

$$\begin{cases} R_o &= \alpha, \\ R_n &= R_{n-1} + \beta n + \gamma, n \geq 1, \end{cases} \qquad R_n = \alpha + \sum_{k=1}^{n} (\beta k + \gamma).$$

Czasem aby obliczyć rekurencję łatwiej jest sprowadzić ją do sumy i obliczyć sumę, a czasem, gdy trudno jest bezpośrednio wyliczyć sumę, można ją sprowadzić do rekurencji i jakimś sposobem rozwiązać rekurencję. $\square$

Problem 2.1.1. Które rekurencje da się sprowadzić do sum? *Na pewno nie można tego zrobić na przykład z taką rekurencją:*

$$\begin{cases} a_o &= \alpha, \\ a_n &= a_{n-1}^2 + 3^n, n \geq 1. \end{cases}$$

Rozważmy jednak rekurencję pojawiającą się w problemie wieży w Hanoi

$$\begin{cases} h_o &= 0, \\ h_n &= 2h_{n-1} + 1, \quad dla \ n \geq 1. \end{cases}$$

Dzieląc pierwsze równanie przez 2^0 *a drugie przez* 2^n *otrzymujemy*

$$\begin{cases} \frac{h_o}{2^0} &= 0, \\ \frac{h_n}{2^n} &= \frac{h_{n-1}}{2^{n-1}} + \frac{1}{2^n}, \quad dla \ n \geq 1, \end{cases}$$

i po podstawieniu $S_n = \frac{h_n}{2^n}$ *otrzymujemy*

$$\begin{cases} S_o &= 0, \\ S_n &= S_{n-1} + \frac{1}{2^n}, \quad dla \ n \geq 1, \end{cases}$$

czyli

$$S_n = \sum_{k=1}^{n} 2^{-k} = 1 - 2^{-n},$$

i wracając do h_n *otrzymujemy*

$$h_n = 2^n - 1, \quad n \geq 1.$$

Przez co należy dzielić "inne" rekurencje aby otrzymać sumy? □

2.2. Metoda czynnika sumacyjnego

Rozważmy rekurencję

$$\begin{cases} a_o T_o &= c_o, \\ a_n T_n &= b_n T_{n-1} + c_n, \text{ dla } n \geq 1, \end{cases} \tag{2.2.1}$$

gdzie $b_n \neq 0, n \geq 1$. Mnożąc obie strony równania (2.2.1) przez s_o i odpowiednio s_n otrzymujemy

$$\begin{cases} a_o s_o T_o &= c_o s_o, \\ \underbrace{a_n s_n T_n} &= \underbrace{b_n s_n T_{n-1}} + c_n s_n, \text{ dla } n \geq 1, \\ S_n &= S_{n-1} + c_n s_n, \quad n \geq 1. \end{cases} \tag{2.2.2}$$

Żeby zapisać tą rekurencję w postaci sumy współczynniki przy T_n muszą nam się "zgadzać" tzn.:

$$s_n b_n = s_{n-1} a_{n-1}, \quad n \geq 1,$$

czyli

$$s_n = s_{n-1}\frac{a_{n-1}}{b_n} = s_{n-2}\frac{a_{n-1}a_{n-2}}{b_n b_{n-1}} = s_o \frac{a_o a_1 ... a_{n-1}}{b_1 b_2 ... b_n}, \quad n \geq 1,$$

i możemy przyjąć $s_o = 1$ otrzymując

$$s_n = \frac{a_o a_1 ... a_{n-1}}{b_1 b_2 ... b_n}, \quad n \geq 1.$$

Wtedy

$$S_n = s_1 b_1 T_o + \sum_{k=1}^{n} s_k c_k, \quad n \geq 0,$$

czyli

$$T_n = \frac{1}{s_n a_n}(s_1 b_1 T_o + \sum_{k=1}^{n} s_k c_k), \quad n \geq 0. \tag{2.2.3}$$

Przykład 2.2.1. *Rozwiąż rekurencję*

$$\begin{cases} \tau_o &= 0, \\ \tau_n &= n+1+\frac{2}{n}\sum_{k=0}^{n-1}\tau_k, \quad dla\ n \geq 1. \end{cases} \tag{2.2.4}$$

Ciąg ten określa średnią liczbę porównań dla jednej z wersji sortowania n losowych elementów metodą "szybkiego sortowania" (quicksort, [10] i [21], str. 95). Szerzej algorytm "szybkiego sortowania" omówimy w rozdziale 7.3.4.

Zwróćmy uwagę, że

$$\begin{array}{lll} & n\tau_n & = n(n+1)+2\sum_{k=0}^{n-1}\tau_k, \quad dla\ n \geq 0, \\ - & (n-1)\tau_{n-1} & = (n-1)n+2\sum_{k=0}^{n-2}\tau_k, \quad dla\ n \geq 1, \\ \hline & n\tau_n-(n-1)\tau_{n-1} & = 2n+2\tau_{n-1}, \quad n \geq 1. \end{array}$$

Czyli rekurencja (2.2.4) jest równoważna

$$\begin{cases} \tau_o &= 0, \\ n\tau_n &= 2n+(n+1)\tau_{n-1}, \quad dla\ n \geq 1. \end{cases} \tag{2.2.5}$$

Stosując metodę czynnika sumacyjnego z $a_n = n\ (n \geq 1), a_o = 1, b_n = n+1\ (n \geq 1), c_n = 2n\ (n \geq 0)$, *wyliczamy czynnik sumacyjny*

$$s_n = \frac{1 \cdot 1 \cdot 2 \cdot (n-1)}{2 \cdot 3 \cdot 4 \cdot ...(n+1)} = \frac{1}{n(n+1)}$$

i stosujemy wzór (2.2.3) otrzymując

$$\begin{aligned} \tau_n &= \frac{1}{s_n a_n}(s_1 b_1 T_o + \sum_{k=1}^{n} s_k c_k) \\ &= \frac{n(n+1)}{n}(\frac{1}{2} \cdot 2 \cdot 0 + \sum_{k=1}^{n} \frac{1}{n(n+1)} \cdot 2k) \\ &= 2(n+1)\sum_{k=1}^{n}\frac{1}{k+1}. \end{aligned}$$

W całym tym skrypcie liczby

$$\sum_{k=1}^{n}\frac{1}{k}, n \geq 1,$$

nazywamy liczbami harmonicznymi i oznaczamy symbolem H_n. *Ważniejsze własności tych liczb podamy w podrozdziale 2.4.*

Jak wyrazić τ_n za pomocą liczb harmonicznych? Korzystamy z przekształceń sum:

$$\begin{aligned}\sum_{k=1}^{n}\frac{1}{n+1} &= \sum_{1\le k\le n}\frac{1}{k+1}\overset{k\to k-1}{=}\sum_{1\le k-1\le n}\frac{1}{k-1+1}\\ &= \sum_{2\le k\le n+1}\frac{1}{k}=\sum_{1\le k\le n}\frac{1}{k}-\frac{1}{1}+\frac{1}{n+1}=H_n-\frac{n}{n+1},\end{aligned}$$

a stąd

$$\tau_n = 2(n+1)H_n - 2n, n \ge 0. \qquad \square$$

2.3. Własności sumowania

Wprowadzony w tym rozdziale operator sumowania może uprościć długie operacje na sumach. Dlatego scharakteryzujmy podstawowe własności tego operatora:

$$\sum_{k\in\mathcal{K}} ca_k = c\sum_{k\in\mathcal{K}} a_k, \qquad \text{Prawo jednorodności,}$$

$$\sum_{k\in\mathcal{K}}(a_k+b_k) = \sum_{k\in\mathcal{K}} a_k + \sum_{k\in\mathcal{K}} b_k, \qquad \text{Prawo addytywności,}$$

$$\sum_{k\in\mathcal{K}} a_k = \sum_{p(k)\in\mathcal{K}} a_{p(k)}, \qquad \text{Prawo przemienności,}$$

$$\sum_{k\in\mathcal{K}} a_k + \sum_{k\in\mathcal{K}'} a_k = \sum_{k\in\mathcal{K}\cup\mathcal{K}'} a_k + \sum_{k\in\mathcal{K}\cap\mathcal{K}'} a_k, \qquad \text{Prawo łączenia zbiorów indeksów sumowania}$$

$$\sum_{k=m}^{n}(a_{k+1}-a_k) = a_{n+1}-a_m, \qquad \text{Prawo składania (“telescoping”),}$$

gdzie $p(k)$ oznacza dowolną permutację liczb całkowitych (np. jak w przykładzie poniżej $p(k)=k+1$, lub $p(k)=-k+c$) o ile powyższe sumy istnieją i są skończone (szeregi). Dokładnie permutacja p jest funkcją taką, że:

$$\begin{aligned}&p : Z \to Z,\\ &\bigwedge_{n\in\mathcal{K}}\bigvee_{k\in Z}!\ p(k)=n.\end{aligned}$$

Prawo przemienności można uogólnić. Niech $f : \mathcal{J} \to \mathcal{K}, \mathcal{J}, \mathcal{K} \subset Z$ będzie dowolnym przekształceniem dwóch zbiorów indeksów. Wtedy

$$\sum_{j\in\mathcal{J}} a_{f(j)} = \sum_{k\in\mathcal{K}} a_k \overline{\overline{f^{-1}(k)}},$$

gdzie $\overline{\overline{f^{-1}(k)}}$ oznacza moc zbioru $\{j : f(j) = k\}$. Konieczność występowania członów $\overline{\overline{f^{-1}(k)}}$ wyjaśnimy na przykładzie:

Przykład 2.3.1. *Niech* $f(k) = k^2, k \in \mathcal{Z}$ *a* $\mathcal{K} = \{-1, 0, 1, 2, 3\}$. *Rozważamy sumę* $\sum_{k\in\mathcal{K}} a_{f(k)}$*:*

$$\begin{aligned}\sum_{k\in\{-1,0,1,2,3\}} a_{f(k)} &= a_{(-1)^2} + a_{0^2} + a_{1^2} + a_{2^2} + a_{3^2} \\ &= a_0 + 2a_1 + a_4 + a_9 \\ &= \sum_{k\in f(\mathcal{K})} a_k \overline{\overline{f^{-1}(k)}}.\end{aligned}$$

□

Przedostatnia tożsamość wynika z

$$[k \in \mathcal{K}] + [k \in \mathcal{K}'] = [k \in \mathcal{K} \cup \mathcal{K}'] + [k \in \mathcal{K} \cap \mathcal{K}']$$

i zapisu sumy w konwencji Iversona.

2.4. Liczby harmoniczne

Dwa następujące wyniki charakteryzują liczby harmoniczne. Pierwszy mówi, że ciąg $\{H_n, n \geq 0\}$ (z definicji sumy $H_0 = \sum_{i=1}^{0} \frac{1}{i} = 0$) jest nieujemny, rosnący, rozbieżny do nieskończoności a drugi, że pełni w matematyce dyskretnej taką rolę, jak funkcja ln(.) w "zwykłej" matematyce. Zaczniemy jednak od tabelki w której wymienimy początkowe liczby harmoniczne:

Tabela 2.1. Liczby harmoniczne H_n.

n	0	1	2	3	4	5	6	7	8	9	10	11
H_n	0	1	$\frac{3}{2}$	$\frac{11}{6}$	$\frac{25}{12}$	$\frac{137}{60}$	$\frac{49}{20}$	$\frac{363}{140}$	$\frac{761}{280}$	$\frac{7129}{2520}$	$\frac{7381}{2520}$	$\frac{83711}{27720}$

Twierdzenie 2.4.1. $\lim_{n\to\infty} H_n = \infty$.

Dowód. Mamy

$$H_{2n} - H_n = \frac{1}{n+1} + \frac{1}{n+2} + ... + \frac{1}{2n} > n \cdot \frac{1}{2n} = \frac{1}{2}$$

i gdyby istniała skończona granica ciągu H_n, to przechodząc w powyższej nierówności stronami do granicy lewa strona byłaby zbieżna do 0 a prawa stale równa $\frac{1}{2}$. □

Twierdzenie 2.4.2. $\lim_{n\to\infty}(H_n - \ln(n)) = \gamma$, *gdzie* $\gamma = 0,57721\,56649\,01533\ldots$ *jest nazywana stałą Eulera.*

Dowód. Rozważmy funkcję $f(x) = \ln(x) - x$. Funkcja ta ma pochodną $f'(x) = \frac{1}{x} - 1$ dodatnią dla $0 < x < 1$ i ujemną dla $x > 1$. Zatem funkcja $f(x)$ osiąga największą wartość dla $x > 0$ w punkcie $x = 1$ równą $f(1) = -1$. Stąd otrzymujemy nierówność

$$\ln(x) \leq x - 1, \quad x \in [0, \infty).$$

A zatem

$$\ln(x+1) \leq x, \quad x \in [-1, \infty).$$

Kładąc $x = \frac{1}{k}$ oraz $x = -\frac{1}{k}$ otrzymujemy

$$\frac{1}{k} \geq \ln(\frac{k+1}{k}) = -\ln(\frac{k}{k+1}) = -\ln(1 - \frac{1}{k+1}) \geq \frac{1}{k+1}, k \geq 1,$$

a więc

$$0 < \frac{1}{k} - \ln(\frac{k+1}{k}) < \frac{1}{k} - \frac{1}{k+1}, k \geq 1.$$

Stąd ciąg $S_n = \sum_{k=1}^{n}(\frac{1}{k} - \ln(\frac{k+1}{k}))$ jest monotoniczny i ograniczony przez $\sum_{k=1}^{n}(\frac{1}{k} - \frac{1}{k+1}) = 1 - \frac{1}{n+1} < 1$, a więc jest zbieżny. Oznaczmy jego granicę przez γ. Zatem z równości

$$\begin{aligned} H_n - \ln(n) &= \sum_{k=1}^{n}(\frac{1}{k} - \ln(\frac{k+1}{k})) + \sum_{k=1}^{n}(\ln(k+1) - \ln(k)) - \ln(n) \\ &= \sum_{k=1}^{n}(\frac{1}{k} - \ln(\frac{k+1}{k})) + \ln(\frac{n+1}{n}) \\ &\to \gamma + \ln(1) \\ &= \gamma. \end{aligned}$$

□

Uwaga 2.4.3. *Nie wiadomo dotąd czy liczba Eulera* γ *jest wymierna czy niewymierna?* □

Uwaga 2.4.4. *Można uzyskać dokładniejsze rozwinięcie:* $H_n \approx \ln(n) + \gamma + \frac{1}{2n} + \frac{1}{12n^2} + \frac{1}{120n^4}$. □

Uwaga 2.4.5. *Liczby harmoniczne spełniają tożsamości*

- $\sum_{i=1}^{n} H_i = (n+1)(H_{n+1} - 1), n \geq 1,$
- $\sum_{i=1}^{n} iH_i = \binom{n+1}{2}(H_{n+1} - \frac{1}{2}), n \geq 1,$
- $\sum_{i=1}^{n} \binom{i}{k} H_i = \binom{n+1}{k+1}(H_{n+1} - \frac{1}{k+1}), k \geq 1.$

Można udowodnić to korzystając z obliczania sum metodą zaburzania (zob. podrozdział 2.6.2). □

2.5. Sumy wielokrotne

W niektórych sytuacjach mamy do czynienia pod znakiem sumy nie z ciągiem $\{a_i, i \geq 1\}$ ale z tablicą $\{a_{i,j}, i, j \geq 1\}$ albo tablicą mającą jeszcze więcej wymiarów. Co prawda wielowymiarowy ciąg zawsze można ustawić w ciąg jednowymiarowy (z teorii mnogości zbiory $\mathcal{N}$ oraz $\mathcal{N}^k$ dla $k \in \mathcal{N}$ są równoliczne - metoda przekątniowa), jednak takie ustawienie w jednowymiarowy ciąg może być skomplikowane i może zaciemnić strukturę badanej sumy. Dlatego czasem rozważa się sumy po wielowymiarowych zbiorach indeksów. Na przykład:

$$\sum_{\substack{1\leq j\leq 3\\ 1\leq k\leq 2}} a_j b_k = a_1b_1 + a_1b_2 + a_2b_1 + a_2b_2 + a_3b_1 + a_3b_2.$$

Podobnie do konwencji Iversona w przypadku dwuwymiarowych sum stosujemy notację

$$\begin{aligned} \sum_{\mathcal{P}(j,k)} a_{j,k} &= \sum_{j,k} a_{j,k}[\mathcal{P}(j,k)], \\ \sum_j \sum_k a_{j,k}[\mathcal{P}(j,k)] &= \sum_{\mathcal{P}(j,k)} a_{j,k} = \sum_k \sum_j a_{j,k}[\mathcal{P}(j,k)]. \end{aligned}$$

Szczególnie interesująca jest druga własność, a zwłaszcza jej dwie wersje:

$$\sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}} a_{j,k} = \sum_{\substack{j\in\mathcal{J}\\ k\in\mathcal{K}}} a_{j,k} = \sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{J}} a_{j,k}, \qquad \text{wersja waniliowa,}$$

$$\sum_{j\in\mathcal{J}} \sum_{k\in\mathcal{K}(j)} a_{j,k} = \sum_{\substack{j\in\mathcal{J}\\ k\in\mathcal{K}(j)}} a_{j,k} = \sum_{k\in\mathcal{K}'} \sum_{j\in\mathcal{J}'(k)} a_{j,k}, \qquad \text{wersja bakaliowa,}$$

oraz szczególny przypadek wersji waniliowej

$$\sum_{\substack{j\in\mathcal{J}\\ k\in\mathcal{K}}} a_j b_k = (\sum_{j\in\mathcal{J}} a_j)(\sum_{k\in\mathcal{K}} b_k).$$

Wersja waniliowa zachodzi wtedy, gdy zbiory indeksów są niezależne

$$[j \in \mathcal{J}, k \in \mathcal{K}] = [j \in \mathcal{J}][k \in \mathcal{K}],$$

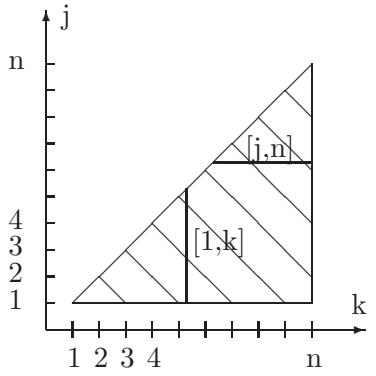
a gdy są zależne to wtedy stosujemy wersję bakaliową z

$$[j \in \mathcal{J}][k \in \mathcal{K}(j)] = [k \in \mathcal{K}'][j \in \mathcal{J}'(k)].$$

Jako przykład wersji bakaliowej rozważmy zbiór $\{(j,k) \in \mathcal{Z}^2 : 1 \le j \le k \le n\}$ lub w konwencji Iversona $[1 \le j \le k \le n]$. Jest to obszar zakreskowany na Rysunku 2.1. Obserwując w jakich granicach globalnie może się j zmieniać widzimy, że $\mathcal{J} = \{1,2,3,...,n\}$ czyli $[1 \le j \le n]$. Ponadto dla dowolnego ustalonego w tym przedziale j zmienność k jest ograniczona do $\mathcal{K}(j) = \{j, j+1, j+2, ..., n\}$ lub w konwencji Iversona $[j \le k \le n]$. Z drugiej strony analizując zmienność k otrzymujemy $\mathcal{K}' = \{1,2,3,...,n\}$ ($[1 \le k \le n]$) a dla dowolnego ustalonego k zmienność j wynosi $\mathcal{J}'(k) = \{1,2,3,...,k\}$ ($[1 \le j \le k]$) a stąd

$$[1 \le j \le n][j \le k \le n] = [1 \le j \le k \le n] = [1 \le k \le n][1 \le j \le k],$$

Rysunek 2.1. Ilustracja wersji bakaliowej przemienności sumowania.



a przechodząc na sumy

$$\sum_{j=1}^{n}\sum_{k=j}^{n} a_{j,k} = \sum_{1\le j\le k\le n} a_{j,k} = \sum_{k=1}^{n}\sum_{j=1}^{k} a_{j,k}.$$

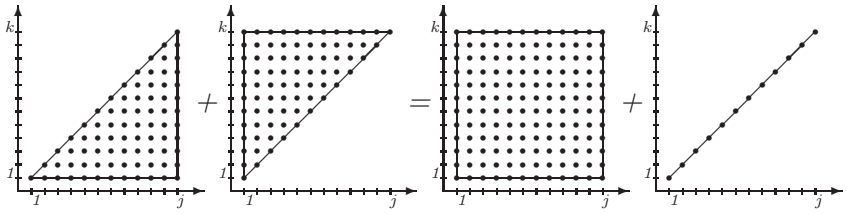
Przykład 2.5.1. *Oblicz sumę górnego trójkąta tablicy*

$$S_\triangledown = \sum_{1\le j\le k\le n} a_j a_k = \sum_{1\le k\le j\le n} a_j a_k = S_\triangle.$$

Ponieważ

$$[1 \le j \le k \le n] + [1 \le k \le j \le n] = [1 \le j \le n][1 \le k \le n] + [1 \le j = k \le n]$$

Rysunek 2.2. Graficzna ilustracja operacji na indeksach sumy.



a więc

$$2S_\nabla = S_\nabla + S_\triangle = (\sum_{j=1}^{n} a_j)^2 + \sum_{j=1}^{n} a_j^2. \qquad \square$$

Przykład 2.5.2. *Oblicz*

$$S = \sum_{1\le j<k\le n} (a_k - a_j)(b_k - b_j).$$

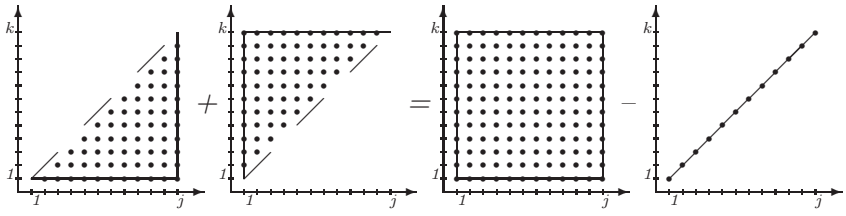
Z symetrii mamy

$$\sum_{1\le j<k\le n} (a_k - a_j)(b_k - b_j) = \sum_{1\le j<k\le n} (a_j - a_k)(b_j - b_k),$$

a z tożsamości

$$[1 \le j < k \le n] + [1 \le k, j \le n] = [1 \le k \le n] - [1 \le j = k \le n]$$

Rysunek 2.3. Graficzna ilustracja operacji na indeksach sumy.



otrzymujemy

$$2S = \sum_{1\le j,k\le n} (a_j - a_k)(b_j - b_k) - \sum_{1\le j=k\le n} (a_j - a_k)(b_j - b_k),$$

i wymnażając czynniki pod znakiem pierwszej sumy (wszystkie o smaku waniliowym) otrzymujemy

$$2S = 2n \sum_{1\le k\le n} a_k b_k - 2(\sum_{k=1}^{n} a_k)(\sum_{k=1}^{n} b_k),$$

a stąd otrzymujemy interesujący wzór

$$(\sum_{k=1}^{n} a_k)(\sum_{k=1}^{n} b_k) = n \sum_{1\le k\le n} a_k b_k - \sum_{1\le j<k\le n} (a_k - a_j)(b_k - b_j). \qquad \square$$

Przykład 2.5.3. *Oblicz*

$$S_n = \sum_{1\le j<k\le n} \frac{1}{2k-j}.$$

Podstawiając $2k - j$ zamiast j wyliczamy następująco sumę S_n:

$$\begin{aligned} S_n &= \sum_{1\le k\le n} (\sum_{1\le j<k} \frac{1}{2k-j}) \overset{j:=2k-j}{=} \sum_{1\le k\le n} (\sum_{1\le 2k-j<k} \frac{1}{2k-2k+j}) \\ &= \sum_{1\le k\le n} \sum_{k<j\le 2k-1} \frac{1}{j} = \sum_{1\le k\le n} H_{2k-1} - \sum_{1\le k\le n} H_k. \end{aligned} \qquad \square$$

Przykład 2.5.4. *Korzystając z poprzedniego wyniku oblicz*

$$\sum_{1\le j^2<k\le n} \frac{1}{k-j^2}. \qquad \square$$

2.6. Różne metody obliczania sum

W tym podrozdziale podamy różne metody rozwiązania następującego problemu:

Problem 2.6.1. *Oblicz*

$$S_n = \sum_{k=1}^{n} k^2, \quad n \ge 1.$$

2.6.1. Metoda indukcyjna

Zgadnij (lub odwiedź jasnowidza), że

$$S_n = \sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}, n \geq 1,$$

i udowodnij wynik indukcyjnie.

Krok 1. Sprawdzamy dla $n = 1$:

$$S_1 = 1 = \frac{1 \cdot 2 \cdot 3}{6}.$$

Krok 2. Załóżmy, że teza jest prawdziwa dla n i dowiedziemy jej prawdziwości dla $n + 1$.

$$\begin{aligned} S_{n+1} &= S_n + (n+1)^2 = \frac{n(n+1)(2n+1) + 6(n+1)^2}{6} \\ &= \frac{(n+1)(2n^2 + n + 6n + 6)}{6} = \frac{(n+1)(2n^2 + 7n + 6)}{6} \\ &= \frac{(n+1)(n+2)(2n+3)}{6}. \end{aligned}$$

2.6.2. Metoda zaburzania

Metoda zaburzania. *Załóżmy, że chcemy obliczyć sumę* $S_n = \sum_{k=1}^{n} a_k$. *Z jednej strony wiemy, że* $S_{n+1} = S_n + a_{n+1}$ *a z drugiej strony* $S_{n+1} = a_1 + \sum_{k=2}^{n+1} a_k$. *Gdyby udało się wyrazić* $\sum_{k=2}^{n+1} a_k$ *za pomocą sumy* $S_n = \sum_{k=1}^{n} a_k$ *(zwróć uwagę że ilość składników w obu sumach jest taka sama), na przykład gdyby* $\sum_{k=2}^{n+1} a_k = f(S_n)$ *to otrzymalibyśmy równanie:*

$$S_n + a_{n+1} = a_1 + f(S_n), n \geq 1,$$

z jedną niewiadomą S_n *którą można by było obliczyć.* □

Zastosujmy metodę zaburzania do naszego przykładu

$$\begin{aligned} S_{n+1} &= S_n + (n+1)^2 = 1^2 + \sum_{k=2}^{n+1} k^2 = 1 + \sum_{k=1}^{n} (k+1)^2 \\ &= 1 + \sum_{k=1}^{n} k^2 + \sum_{k=1}^{n} (2k+1) = 1 + S_n + 2\sum_{k=1}^{n} k + n \\ &= S_n + 2\sum_{k=1}^{n} k + (n+1), \end{aligned}$$

a stąd

$$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}, \quad n \geq 1.$$

Metoda zaburzania nie doprowadziła nas do celu, poszukiwana zmienna S_n zredukowała się po obu stronach równania! Niechcący zaburzenie sumy $S_n = \sum_{k=1}^{n} k^2$ pozwoliło obliczyć sumę $\sum_{k=1}^{n} k$, więc wydaje się, że aby znaleźć naszą sumę, powinniśmy zaburzyć sumę $T_n = \sum_{k=1}^{n} k^3, \quad n \geq 1$. Postępując tak otrzymujemy

$$\begin{aligned} T_{n+1} &= T_n + (n+1)^3 = 1^3 + \sum_{k=2}^{n+1} k^3 = 1 + \sum_{k=1}^{n} (k+1)^3 \\ &= 1 + \sum_{k=1}^{n} k^3 + 3\sum_{k=1}^{n} k^2 + 3\sum_{k=1}^{n} k + n \\ &= 1 + T_n + 3S_n + 3\frac{n(n+1)}{2} + n, \end{aligned}$$

a stąd

$$(n+1)^3 = 3S_n + \frac{3n(n+1)}{2} + (n+1),$$

co prowadzi do

$$\frac{n(n+1)(2n+1)}{6} = S_n, \quad n \geq 1.$$

Podajmy inny przykład metody zaburzania, obliczmy sumę $S_n = \sum_{k=1}^{n} 2^k$. Analizując tę sumę otrzymujemy

$$S_{n+1} = S_n + 2^{n+1} = 1 + \sum_{k=2}^{n+1} 2^k = 1 + \sum_{k=1}^{n} 2^{k+1} = 1 + 2\sum_{k=1}^{n} 2^k = 1 + 2S_n, \; n \geq 1,$$

a więc

$$S_n + 2^{n+1} = 1 + 2S_n, \; n \geq 1,$$

co prowadzi do

$$S_n = \sum_{k=1}^{n} 2^k = 2^{n+1} - 1,$$

(por. wzór na sumę ciągu geometrycznego Przykład 1.2.2).

Problem 2.6.2. *Oblicz metodą zaburzania sumę*

$$(a) \; S_n = \sum_{0 \leq k \leq n} kx^k, \quad x \neq 1,$$

$$\begin{aligned} (b)\ S_n &= \sum_{1\le k\le n} H_k, (\text{zaburz sumę } \sum_{1\le k\le n} kH_k), \\ (c)\ S_n &= \sum_{1\le k\le n} sin(kx). \end{aligned}$$ □

2.6.3. Przejdź do rekurencji i skonstruuj repertuar

Zapisując tę sumę w formie rekurencji otrzymujemy

$$\begin{cases} S_1 &= 1, \\ S_n &= S_{n-1} + n^2 \ \text{ dla } \ n > 1, \end{cases}$$

którą uogólniamy do

$$\begin{cases} S(1) &= \alpha, \\ S(n) &= S(n-1) + \beta n^2 + \gamma n + \delta \ \text{ dla } \ n \ge 1, \end{cases}$$

i szukamy rozwiązania w postaci

$$S(n) = \alpha A(n) + \beta B(n) + \gamma C(n) + \delta D(n), \ \ n \ge 1.$$

Kolejno dla funkcji $S(n) = 1, S(n) = n, S(n) = n^2, S(n) = n^3$, otrzymujemy rozwiązania

$$\begin{aligned} &\alpha = 1, \beta = 0, \gamma = 0, \delta = 0, \\ &\alpha = 1, \beta = 0, \gamma = 0, \delta = 1, \\ &\alpha = 1, \beta = 0, \gamma = 2, \delta = -1, \\ &\alpha = 1, \beta = 3, \gamma = -3, \delta = 1, \end{aligned}$$

czyli otrzymujemy układ równań

$$\begin{aligned} 1 &= A(n), \\ n &= A(n) + D(n), \\ n^2 &= A(n) + 2C(n) - D(n), \\ n^3 &= A(n) + 3B(n) - 3C(n) + D(n), \end{aligned}$$

co można rozwiązać

$$\begin{aligned} A(n) &= 1, \\ B(n) &= \frac{2n^3 + 3n^2 + n - 6}{6}, \end{aligned}$$

$$\begin{aligned} C(n) &= \frac{n^2+n-2}{2}, \\ D(n) &= n-1, \end{aligned}$$

a więc otrzymujemy ogólne rozwiązanie

$$S(n) = \alpha + \beta\frac{2n^3+3n^2+n-6}{6} + \gamma\frac{n^2+n-2}{2} + \delta(n-1),$$

ze szczególnym przypadkiem $\alpha = \beta = 1, \gamma = \delta = 0$,

$$S_n = 1 + \frac{2n^3+3n^2+n-6}{6} = \frac{n(n+1)(2n+1)}{6}, \quad n \geq 1.$$

Uwaga!!! Inne metody obliczania sum to metody różnicowe i metoda funkcji tworzących, które poznamy dalej.

2.7. Rachunek różnicowy

Rachunek różnicowy powstał przez analogię do rachunku różniczkowego - działu matematyki (analiza matematyczna) zajmującego się badaniem funkcji rzeczywistych i zespolonych, przy użyciu ich pochodnych i całek.

2.7.1. Podstawowe pojęcia

Podstawą rachunku różniczkowego jest operator pochodnej D, zdefiniowany jako

$$(Df)(x) = \lim_{h\to 0}\frac{f(x+h)-f(x)}{h}.$$

Przyporządkowuje on funkcję Df funkcji rzeczywistej f. Czasem zamiast zapisu $(Df)(x)$ stosuje się $f'(x)$ lub $\frac{\partial f}{\partial x}(x)$. Odpowiednikiem operatora pochodnej w rachunku różnicowym jest operator różnicowy $\triangle$, zdefiniowany jako

$$(\triangle f)(x) = f(x+1) - f(x).$$

Przyporządkowuje on funkcję $\triangle f$ funkcji rzeczywistej f. Będziemy go jednak rozważać tylko dla funkcji określonych na zbiorze liczb naturalnych (czyli dla ciągów). Operator $\triangle$ to "skończony odpowiednik" operatora D. Rozważając funkcję liczb całkowitych f nie mamy możliwości badać granicy występującej w definicji D. W zamian za to rozważamy stosowny iloraz $\frac{f(x+1)-f(x)}{1}$ przy najmniejszej możliwej naturalnej wartości h.

Operator $\triangle$ możemy iterować:

$$\begin{aligned} \triangle^0 f &= f, \\ \triangle^n f &= \triangle(\triangle^{n-1} f), \ \ n \geq 1. \end{aligned}$$

Na przykład jeśli $f(x) = \sum_{k=1}^{x} k^2, x \geq 1$, to

$$\begin{aligned} \triangle^0 f &= \sum_{k=1}^{x} k^2 = \frac{x(x+1)(2x+1)}{6}, \\ \triangle f &= (x+1)^2, \\ \triangle^2 f &= 2x+1, \\ \triangle^3 f &= 2, \\ \triangle^4 f &= 0. \end{aligned}$$

2.7.2. Własności operatora różnicowego

Uwaga 2.7.1. *Operator różnicowy spełnia następujące własności:*

$$\begin{aligned} \triangle(cf(x)) &= c \cdot \triangle f(x), \\ \triangle(f+g)(x) &= \triangle f(x) + \triangle g(x), \\ \triangle^m x^n &= 0, \ \textit{dla } m > n, \ \triangle^n x^n = n!, \\ \triangle^n f(x) &= \sum_{k=0}^{n} (-1)^k \binom{n}{k} f(x+n-k), \\ f(x+n) &= \sum_{k=0}^{n} \binom{n}{k} \triangle^k f(x), \end{aligned}$$

dla dowolnej stałej c, dowolnych funkcji f i g oraz naturalnych n i m. □

Dwie pierwsze własności wykazujemy bezpośrednio z definicji a trzy ostatnie dowodzimy indukcyjnie.

Szczególnie łatwo jest obliczyć pochodną D dla funkcji postaci $f(x) = x^a$. Mamy bowiem $D(x^a) = ax^{a-1}, a \neq 0$. Niestety, nie jest ten wzór prawdziwy dla operatora $\triangle$. Na przykład $\triangle(x^2) = (x+1)^2 - x^2 = 2x+1$, $\triangle(x^3) = (x+1)^3 - x^3 = 3x^2 + 3x + 1$. Istnieje jednak funkcja która daje podobne wzory dla operatora $\triangle$.

Definicja 2.7.1. *Zdefiniujmy funkcję m-ta dolna silnia od x przez*

$$x^{\underline{m}} = \begin{cases} x(x-1)(x-2)...(x-m+1), & \textit{dla } m > 0, \\ 1, & \textit{dla } m = 0, \\ \frac{1}{(x+1)(x+2)(x+3)...(x+(-m))}, & \textit{dla } m < 0. \end{cases}$$ □

Kolejne wartości tej funkcji w zależności od zwykłych potęg przedstawiamy w tabeli:

Tabela 2.2. Tablica potęg i dolnych silni.

$k^{\underline{0}} = 1$	$k^0 = 1$
$k^{\underline{1}} = k$	$k^1 = k^{\underline{1}}$
$k^{\underline{2}} = k^2 - k$	$k^2 = k^{\underline{2}} + k^{\underline{1}}$
$k^{\underline{3}} = k^3 - 3k^2 + 2k$	$k^3 = k^{\underline{3}} + 3k^{\underline{2}} + k^{\underline{1}}$
$k^{\underline{4}} = k^4 - 6k^3 + 11k^2 - 6k$	$k^4 = k^{\underline{4}} + 6k^{\underline{3}} + 7k^{\underline{2}} + k^{\underline{1}}$
$k^{\underline{5}} = k^5 - 10k^4 + 35k^3 - 50k^2 + 24k$	$k^5 = k^{\underline{5}} + 10k^{\underline{4}} + 25k^{\underline{3}} + 15k^{\underline{2}} + k^{\underline{1}}$

Twierdzenie 2.7.2. *Dla dowolnych liczb całkowitych $m \neq 0, a, n$ zachodzi*

$$\begin{aligned}
\triangle(x+a)^{\underline{m}} &= m(x+a)^{\underline{m-1}},\\
\triangle H_{x-1} &= \frac{1}{x},\\
\triangle a^x &= (a-1)\cdot a^x,\\
\triangle\binom{x}{n} &= \binom{x}{n-1},\\
\triangle x^n &= \sum_{k=1}^{n}\binom{n}{k}x^{n-k},\\
\triangle \sin(x) &= 2\sin(\frac{1}{2})\cos(x+\frac{1}{2}),\\
\triangle \cos(x) &= -2\sin(\frac{1}{2})\sin(x+\frac{1}{2}).
\end{aligned}$$

Dowód. Dla liczby naturalnej m

$$\begin{aligned}
\triangle(x+a)^{\underline{m}} &= (x+a+1)(x+a)...(x+a-m+2)\\
&\quad -(x+a)(x+a-1)...(x+a-m+1)\\
&= (x+a)(x+a-1)...(x+a-m+2)(x+a+1-(x+a-m+1))\\
&= m(x+a)^{\underline{m-1}},
\end{aligned}$$

natomiast

$$\begin{aligned}
\triangle(x+a)^{\underline{-m}} &= \frac{1}{(x+a+2)...(x+a+m+1)} - \frac{1}{(x+a+1)...(x+a+m)}\\
&= \frac{1}{(x+a+2)...(x+a+m)}\left(\frac{1}{x+a+m+1} - \frac{1}{x+a+1}\right)
\end{aligned}$$

$$\begin{aligned} &= \frac{1}{(x+a+2)...(x+a+m)} \cdot \frac{x+a+1-(x+a+m+1)}{(x+a+1)(x+a+m+1)} \\ &= \frac{1}{(x+a+1)...(x+a+m+1)} \cdot (-m) \\ &= (-m)(x+a)^{\underline{-m-1}}. \end{aligned}$$

Druga równość wynika natychmiast z definicji ciągu liczb harmonicznych, trzecia jest oczywista, pozostałe możemy dowieść bezpośrednio z definicji. □

Warto zauważyć, że z powyższego twierdzenia wynika, że $\triangle 2^x = 2^x$, a więc funkcja 2^x odgrywa taką role w rachunku różnicowym jak funkcja e^x w rachunku różniczkowym. Natomiast funkcja H_x odgrywa taką rolę jak $\ln(x)$, co zauważyliśmy już w podrozdziale 2.4.

2.7.3. Operator różnicowy $\triangle$ a operator sumy $\sum$

Operatorem odwrotnym do operatora D jest operator całki $\int$. Podobnie można rozważać operator odwrotny do $\triangle$ jako operator $\sum$,

$$\begin{aligned} Df(x) = g(x) &\iff f(x) = \int g(x)dx + C, \\ \triangle f(x) = g(x) &\iff f(x) = \sum g(x)\delta x + C. \end{aligned}$$

Uwaga 2.7.3.

$$\begin{aligned} \sum (cf(x))\delta x &= c\sum f(x)\delta x, \\ \sum (f(x)+g(x))\delta x &= c\sum f(x)\delta x + \sum g(x)\delta x. \end{aligned}$$ □

W przypadku całek operator δ oznacza najmniejszy możliwy przyrost, podobnie jest i w przypadku sum, tutaj najmniejszy możliwy przyrost $\delta x = (x+1) - x = 1$.

W analizie rozważa się zarówno całki nieoznaczone jak i oznaczone. Podobnie możemy postępować w przypadku rachunku różnicowego:

$$\begin{aligned} Df(x) = g(x) &\iff f(b) - f(a) = \int_a^b g(x)dx, \\ \triangle f(x) = g(x) &\iff f(b) - f(a) = \sum_a^b f(x) = \sum_{a \le x < b} g(x) = \sum_{x=a}^{b-1} g(x). \end{aligned}$$

Z Twierdzenia 2.7.2 bezpośrednio otrzymujemy

Twierdzenie 2.7.4. *Dla dowolnej liczby całkowitej m oraz naturalnych n i rzeczywistego $a \neq 1$ mamy*

$$\begin{aligned}\sum_{k=0}^{n-1} k^{\underline{m}} &= \begin{cases} \frac{1}{m+1}(n^{\underline{m+1}} - 0^{\underline{m+1}}) & \text{dla } m \neq -1, \\ H_n & \text{dla } m = -1, \end{cases} \\ \sum_{k=0}^{n-1} a^k &= \frac{a^n - 1}{a-1}. \end{aligned}$$

□

Uwaga 2.7.5. *Twierdzenie 2.7.2 pozwala nam uzyskać szereg bardzo interesujących zależności, na przykład:*

$$\begin{aligned}\sum_{k=n}^{2n-1} \binom{k}{n-1} &= \binom{2n}{n} - 1, \\ \sum_{k=0}^{n-1} \cos(k+\frac{1}{2}) &= \frac{\sin(n)}{2\sin(\frac{1}{2})}, \quad \sum_{k=0}^{n-1} \sin(k+\frac{1}{2}) = \frac{\cos(n)}{2\sin(\frac{1}{2})}. \end{aligned}$$

□

2.7.4. Obliczanie sum metodami różnicowymi

Wróćmy do naszego przykładu $S_n = \sum_{k=1}^{n} k^2 = \sum_{k=0}^{n} k^2$. Z Tablicy 2.2 i Twierdzenia 2.7.4 otrzymujemy

$$\begin{aligned} S_{n-1} &= \sum_{0 \leq k < n} k^2 = \sum_{0 \leq k < n} k^{\underline{2}} + \sum_{0 \leq k < n} k^{\underline{1}} = \frac{1}{3} n^{\underline{3}} + \frac{1}{2} n^{\underline{2}} \\ &= \frac{n(n-1)(n-2)}{3} + \frac{n(n-1)}{2} = \frac{n(n-1)(2n-1)}{6}, \end{aligned}$$

i zamieniając $n-1$ na n otrzymujemy tezę.

Przykład 2.7.1. *Obliczyć sumę: $\sum_{k=0}^{n-1}(k^3 - 3k + 2^k + \frac{k^3-1}{(k+1)(k+2)(k+3)})$. Ze wzorów z Tabeli 2.2*

$$\begin{aligned} k^3 &= k^{\underline{3}} + 3k^{\underline{2}} + k^{\underline{1}}, \\ k &= k^{\underline{1}}. \end{aligned}$$

Postaramy się najpierw wyrazić $\frac{k^3-1}{(k+1)(k+2)(k+3)}$ za pomocą dolnych silni. Procedura jest taka, weź wyraz z największą potęgą w liczniku - u nas k^3 (ze współczynnikiem liczbowym) i zmniejsz tą potęge o 1 (k^2) oraz ostatni czynnik w mianowniku - tutaj $(k+3)$ i zastąp w liczniku ten wyraz z najwyższą potęgą k^3 przez iloczyn $k^2(k+3)$ a dalej uzupełnij licznik tak, aby jego wartość się nie zmieniła ($k^3 - 1 = k^2(k+3) - 3k^2 - 1$) rozbij na dwa

ułamki i postępuj tak dalej. Wszystkie potrzebne przekształcenia podajemy poniżej

$$\begin{aligned}
\frac{k^3-1}{(k+1)(k+2)(k+3)} &= \frac{k^3+3k^2-3k^2-1}{(k+1)(k+2)(k+3)} \\
&= \frac{k^2(k+3)}{(k+1)(k+2)(k+3)} - \frac{3k^2+1}{(k+1)(k+2)(k+3)} \\
&= \frac{k^2}{(k+1)(k+2)} - \frac{3k^2+9k-9k+1}{(k+1)(k+2)(k+3)} \\
&= \frac{k(k+2)}{(k+1)(k+2)} - \frac{2k}{(k+1)(k+2)} \\
&\quad -\frac{3k(k+3)}{(k+1)(k+2)(k+3)} + \frac{9k-1}{(k+1)(k+2)(k+3)} \\
&= \frac{k}{k+1} - \frac{2k+4-4}{(k+1)(k+2)} \\
&\quad -\frac{3k+6-6}{(k+1)(k+2)} + \frac{9k+27-28}{(k+1)(k+2)(k+3)} \\
&= \frac{k+1-1}{k+1} - \frac{2(k+2)}{(k+1)(k+2)} \\
&\quad -\frac{4}{(k+1)(k+2)} - \frac{3(k+2)}{(k+1)(k+2)} \\
&\quad +\frac{6}{(k+1)(k+2)} + \frac{9k+27-27-1}{(k+1)(k+2)(k+3)} \\
&= 1 - k^{\underline{-1}} - 2k^{\underline{-1}} + 4k^{\underline{-2}} - 3k^{\underline{-1}} + 6k^{\underline{-2}} + 9k^{\underline{-2}} - 28k^{\underline{-3}} \\
&= k^{\underline{0}} - 6k^{\underline{-1}} + 19k^{\underline{-2}} - 28k^{\underline{-3}}.
\end{aligned}$$

Tak więc otrzymaliśmy sumy składników, które potrafimy obliczyć metodami różnicowymi (z Twierdzenia 2.7.4).

$$\begin{aligned}
&\sum_{k=0}^{n-1}(k^3-3k+2^k+\frac{k^3-1}{(k+1)(k+2)(k+3)}) \\
&= \sum_{k=0}^{n-1}\Big(2^k + k^{\underline{3}} + 3k^{\underline{2}} + 2k^{\underline{1}} + k^{\underline{0}} - 6k^{\underline{-1}} + 19k^{\underline{-2}} - 28k^{\underline{-3}}\Big) \\
&= 2^k\,|_0^n + \frac{k^{\underline{4}}}{4}\,|_0^n + \frac{3k^{\underline{3}}}{3}\,|_0^n + \frac{2k^{\underline{2}}}{2}\,|_0^n + \frac{k^{\underline{1}}}{1}\,|_0^n - 6H_k\,|_0^n + \frac{19k^{\underline{-1}}}{-1}\,|_0^n - \frac{28k^{\underline{-2}}}{-2}\,|_0^n \\
&= 2^n - 1 + \frac{n^{\underline{4}}}{4} + 3\frac{n^{\underline{3}}}{3} + 2\frac{n^{\underline{2}}}{2} + \frac{n^{\underline{1}}}{1} - 6H_n + 19(\frac{n^{\underline{-1}}}{-1} - 1) - 28(\frac{n^{\underline{-2}}}{-2} - \frac{1}{(-2)2!}) \\
&= 2^n - 27 + \frac{n^4-6n^3+11n^2-6n}{4} + n^3 - 3n^2 + 2n + n^2 - n + n
\end{aligned}$$

$$\begin{aligned} & -6H_n - \frac{19}{n+1} + \frac{14}{(n+1)(n+2)} \\ = \; & 2^n + \frac{n^4}{4} - \frac{n^3}{2} + \frac{3n^2}{4} + \frac{n}{2} - 27 - 6H_n - \frac{19}{n+1} + \frac{14}{(n+1)(n+2)}. \end{aligned}$$ □

Uwaga 2.7.6. *Jeżeli mamy obliczyć iloraz różnicowy ułamka, w którym nie ma w mianowniku ciągu arytmetycznego (z $r = 1$) liczb, np. $\frac{2k^2-3}{(k+2)(k+5)}$, to wtedy uzupełniamy mianownik i licznik domnażając przez odpowiednie wyrazy $\frac{(2k^2-3)(k+1)(k+3)(k+4)}{(k+1)(k+2)(k+3)(k+4)(k+5)}$.* □

Uwaga 2.7.7. *Reguły postępowania przy obliczaniu sum metodami różnicowymi to:*

- ♣ *Wyraź wszystkie składniki sumy za pomocą dolnych silni $k^{\underline{m}}$ (Tabela 2.2), potęg a^k lub symboli Newtona $\binom{k}{n}$ lub, jak zobaczymy w następnym podrozdziale iloczynów opisanych tutaj typów.*
- ♣ *Zastosuj Twierdzenie 2.7.4, Twierdzenie 2.7.3, dla $\binom{x}{n}$ lub podany w następnym podrozdziale wzór na różnicowanie przez części dla iloczynów i Twierdzeń 2.7.3-2.7.4.*
- ♣ *Znowu stosując Tabelę 2.2 lub wyliczając bezpośrednio zamień dolną silnię na zwykłe potęgi i uporządkuj wynik.* □

2.7.5. Wzór na różnicę iloczynu i sumowanie przez części

Wprowadźmy najpierw operator przesunięcia E poprzez $Ef(x) = f(x+1)$. Oczywiście stąd $E^n f(x) = f(x+n), n \geq 1$. W rachunku różniczkowym znane są wzory na pochodną iloczynu i ilorazu funkcji:

$$\begin{aligned} D(f \cdot g)(x) &= (Df)(x)g(x) + f(x)(Dg)(x), \qquad (f \cdot g)' = f'g + fg', \\ D(\frac{f}{g})(x) &= \frac{(Df)(x)g(x) - f(x)(Dg)(x)}{g^2(x)}, \qquad (\frac{f}{g})' = \frac{f'g - fg'}{g^2}. \end{aligned}$$

Czy podobne wzory można również podać w rachunku różnicowym? Odpowiedź jest pozytywna, jednak pojawiają się pewne drobne różnice.

Twierdzenie 2.7.8. *Dla dowolnych całkowitoliczbowych funkcji*

$$\begin{aligned} \triangle(f \cdot g)(x) &= f(x)(\triangle g)(x) + (\triangle f)(x)g(x+1) = (f(\triangle g) + (\triangle f)(Eg))(x), \\ \triangle(\frac{f}{g})(x) &= \frac{(\triangle f)(x)g(x) - f(x)(\triangle g)(x)}{g(x)g(x+1)} = \frac{(\triangle f)g - f(\triangle g)}{gEg}(x). \end{aligned}$$

Dowód. Z definicji operatora różnicowego i E otrzymujemy

$$\triangle(f \cdot g)(x) = f(x+1)g(x+1) - f(x)g(x)$$

$$\begin{aligned} &= f(x+1)g(x+1) - f(x)g(x+1) + f(x)g(x+1) - f(x)g(x) \\ &= g(x+1)(\triangle f)(x) + f(x)(\triangle g)(x) \\ &= (f(\triangle g) + (\triangle f)(Eg))(x), \end{aligned}$$

oraz

$$\begin{aligned} \triangle(\frac{f}{g})(x) &= \frac{f(x+1)}{g(x+1)} - \frac{f(x)}{g(x)} \\ &= \frac{f(x+1)g(x) - f(x)g(x+1)}{g(x)g(x+1)} \\ &= \frac{(f(x+1) - f(x))g(x) - f(x)(g(x+1) - g(x))}{g(x)g(x+1)} \\ &= \frac{(\triangle f)g - f(\triangle g)}{gEg}(x). \end{aligned}$$

□

Twierdzenia Leibnitza pomagające obliczyć n-tą pochodną iloczynu funkcji:

$$(fg)^{(n)} = \sum_{k=1}^{n} \binom{n}{k} f^{(k)} g^{(n-k)},$$

gdzie $f^{(n)}$ oznacza n-tą pochodną funkcji f. Wynik analogiczny w rachunku różnicowym do tego twierdzenia można uzyskać indukcyjnie z Twierdzenia 2.7.8.

Twierdzenie 2.7.9. *Zachodzi*

$$\triangle^n(f\cdot g)(x) = \sum_{k=0}^{n} \binom{n}{k} (\triangle^k f)(x)(\triangle^{n-k} g)(x+k) = \sum_{k=0}^{n} \binom{n}{k} ((\triangle^k f)(\triangle^{n-k}(E^k g)))(x).$$

□

Twierdzenie 2.7.8 pozwala uzyskać ciekawy wzór na różnicowanie przez części, poprzez analogię do całkowania przez części

$$\int f(x)g'(x)dx = f(x)g(x) - \int f'(x)g(x)dx.$$

Twierdzenie 2.7.10.

$$\sum f(x) \cdot \triangle g(x)\delta x = f(x) \cdot g(x) - \sum (\triangle f)(x)g(x+1)\delta x. \qquad \square$$

Pokażemy jak ten wzór można wykorzystać do obliczenia sumy $\sum_{0\le k<n} k2^k$. Oczywiście mamy $\triangle 2^k = 2^k$ więc

$$\begin{aligned}
\sum_{0\le k<n} k2^k &= \sum_{0\le k<n} k\triangle 2^k \\
&= n2^n - \sum_{0\le k<n} \triangle k \underline{1} 2^{k+1} \\
&= n2^n - \sum_{0\le k<n} 2^{k+1} \\
&= n2^n - \sum_{0\le k<n+1} 2^k + 1 \\
&= n2^n - 2^{n+1} + 1 + 1 \\
&= n2^n - 2^{n+1} + 2.
\end{aligned}$$

Podsumowanie.

- **Sumy** są szczególnymi rodzajami rekurencji i niektóre rekurencje dają się sprowadzić do sum i obliczyć metodami charakterystycznymi dla operacji na sumach. Sprowadzanie rekurencji do sum nazywa się obliczaniem rekurencji metodą czynnika sumacyjnego.
- Liczby harmoniczne są to sumy odwrotności kolejnych liczb naturalnych. Oznaczamy je przez H_n i odgrywają one ważną rolę w matematyce dyskretnej i przyrodzie (podobną do roli logarytmu naturalnego w analizie).
- Sumy obliczamy metodą zaburzania, poprzez próbę wyrażenia sumy składników od wyrazu 2 do $n+1$ jako suma składników od wyrazu 1 do n i potem rozwiązania równania. Jednak często nie udaje nam się tak wyrazić tych sum a po drugie często otrzymujemy równanie uwikłane, którego nie potrafimy rozwiązać. Drugim potężnym narzędziem obliczania sum jest rachunek różnicowy, który pozwala nam wyliczyć wiele interesujących sum.
- Sumy wielokrotne często daje się obliczyć poprzez geometryczną analizę zbioru indeksowego i rozpisanie tego zbioru w postaci sum i/lub iloczynów innych zbiorów.

Rozdział 3

Funkcje całkowitoliczbowe

3.1. Podłoga i sufit **42**
3.2. Widmo liczby **46**
3.3. Rekurencje z funkcjami podłoga i sufit **47**
3.4. Działanie dwuargumentowe *mod* **52**
3.5. Generatory liczb pseudolosowych **54**
3.6. Zapisy liczb naturalnych w różnych podstawach liczenia **56**

Funkcje całkowitoliczbowe są to funkcję, których zbiorem wartości jest zbiór liczb całkowitych lub jego podzbiór. W tym rozdziale rozważamy funkcje określone na zbiorze liczb rzeczywistych. W bibliotece JAVA - java.lang.Math występują funkcje: $ceil(), floor(), rint(), round()$ oraz operatory % i $x \, mod \, y$, podobnie w języku PYTHON w bibliotece matematycznej występują trzy takie funkcje: $floor(,)$ $ceil()$ i $divmod()$ oraz operatory % i \\. W tym rozdziale będziemy zajmować się takimi właśnie funkcjami.

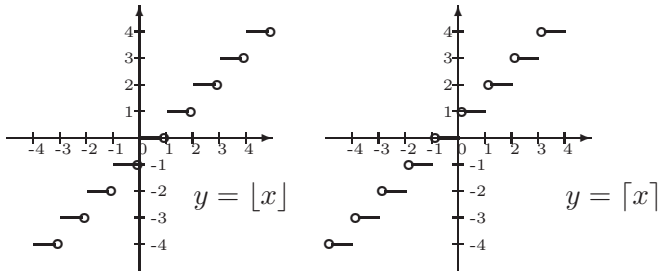
3.1. Podłoga i sufit

Definicja 3.1.1. *Funkcje podłoga* $\lfloor . \rfloor$ *oraz sufit* $\lceil . \rceil$ *definiujemy wzorami:*

$$\begin{aligned} \lfloor x \rfloor &= \max\{c \in Z : c \leq x\}, \\ \lceil x \rceil &= \min\{c \in Z : c \geq x\}. \end{aligned}$$

□

Rysunek 3.1. Funkcje "podłoga" i "sufit".



W językach programowania funkcji podłoga najczęściej odpowiada funkcja $floor()$ natomiast sufit to $ceil()$. W VISUAL BASICu funkcja $INT()$ działa jak podłoga ale jest też inna funkcja, $FIX()$, określona następująco

$$FIX(x) = \begin{cases} \lfloor x \rfloor, & \text{dla } x \geq 0, \\ \lceil x \rceil, & \text{dla } x < 0. \end{cases}$$

Zwróćmy uwagę na interesujący związek funkcji podłoga i sufit z omawianym w poprzednim rozdziale symbolem Iversona i zbiorem liczb całkowitych

$$\lceil x \rceil - \lfloor x \rfloor = [x \text{ nie jest liczbą całkowitą}] = \begin{cases} 0, & x \text{ jest liczbą całkowitą}, \\ 1, & x \text{ nie jest liczbą całkowitą}. \end{cases}$$

Ponadto bezpośrednio z definicji zachodzą następujące nierówności:

$$x-1<\lfloor x\rfloor\leq x\leq\lceil x\rceil<x+1,$$
$$\lfloor -x\rfloor=-\lceil x\rceil,\quad \lceil -x\rceil=-\lfloor x\rfloor.$$

Z tych prostych własności bezpośrednio można otrzymać:

Uwaga 3.1.1. *Zachodzi*

$$\begin{aligned}\lfloor x\rfloor=n &\iff n\leq x<n+1,\\ \lfloor x\rfloor=n &\iff x-1<n\leq x,\\ \lceil x\rceil=n &\iff n-1<x\leq n,\\ \lceil x\rceil=n &\iff x\leq n<x+1,\end{aligned}$$

a ponadto

$$\begin{aligned}x<n &\iff \lfloor x\rfloor<n,\\ n<x &\iff n<\lceil x\rceil,\\ x\leq n &\iff \lceil x\rceil\leq n,\\ n\leq x &\iff n\leq\lfloor x\rfloor.\end{aligned}$$ □

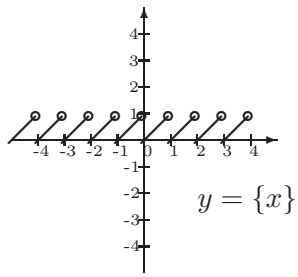
Problem 3.1.1. *Czy*

$$\lfloor x+y\rfloor=\lfloor x\rfloor+\lfloor y\rfloor,\qquad \lceil x+y\rceil=\lceil x\rceil+\lceil y\rceil,$$
$$\lfloor x\cdot y\rfloor=\lfloor x\rfloor\cdot\lfloor y\rfloor,\qquad \lceil x\cdot y\rceil=\lceil x\rceil\cdot\lceil y\rceil?$$

Łatwo sprawdzić, że dla naturalnych liczb n: $\lfloor x\pm n\rfloor=\lfloor x\rfloor\pm n$ *i* $\lceil x\pm n\rceil=\lceil x\rceil\pm n$, *ale w ogólności nie jest prawdziwe* $\lfloor nx\rfloor=n\lfloor x\rfloor$, *czy* $\lceil nx\rceil=n\lceil x\rceil$, *(połóż* $n=2, x=\frac{1}{2}$*).*

Zdefiniujmy $\{x\}=x-\lfloor x\rfloor$ *(jest to część ułamkowa x). Przebieg tej funkcji ilustruje Rys. 3.2*

Rysunek 3.2. Funkcja $\{x\}$.



Z drugiej strony mamy $x = \{x\} + \lfloor x \rfloor$ *a stąd*

$$\lfloor x+y \rfloor = \lfloor \lfloor x \rfloor + \lfloor y \rfloor + \{x\} + \{y\} \rfloor = \lfloor x \rfloor + \lfloor y \rfloor + \lfloor \{x\} + \{y\} \rfloor,$$

a ponieważ $0 \leq \{x\} + \{y\} < 2$, *więc*

$$\lfloor x+y \rfloor = \begin{cases} \lfloor x \rfloor + \lfloor y \rfloor, & gdy\ 0 \leq \{x\} + \{y\} < 1, \\ \lfloor x \rfloor + \lfloor y \rfloor + 1, & gdy\ 1 \leq \{x\} + \{y\} < 2. \end{cases}$$ □

Twierdzenie 3.1.2. *Dla dowolnej nieujemnej rzeczywistej liczby* x *zachodzi*

$$\begin{aligned} \left\lfloor \sqrt{\lfloor x \rfloor} \right\rfloor &= \left\lfloor \sqrt{x} \right\rfloor, \\ \left\lceil \sqrt{\lceil x \rceil} \right\rceil &= \left\lceil \sqrt{x} \right\rceil. \end{aligned}$$ □

Udowodnimy twierdzenie ogólniejsze:

Twierdzenie 3.1.3. *Niech* f *będzie rosnącą ciągłą funkcją o tej własności, że*

Jeśli $f(x)$ *jest liczbą całkowitą to* x *jest liczbą całkowitą*

Wtedy

$$\begin{aligned} \left\lfloor f(\lfloor x \rfloor) \right\rfloor &= \left\lfloor f(x) \right\rfloor, \\ \left\lceil f(\lceil x \rceil) \right\rceil &= \left\lceil f(x) \right\rceil. \end{aligned}$$

Dowód. Udowodnimy tylko pierwszą równość (druga jest analogiczna). Jeżeli $\lfloor x \rfloor = x$ to teza jest oczywista, załóżmy więc $\lfloor x \rfloor < x$. Z tego, że funkcja f jest rosnąca wynika, że $f(\lfloor x \rfloor) < f(x)$ a z własności funkcji podłoga $\left\lfloor f(\lfloor x \rfloor) \right\rfloor \leq \left\lfloor f(x) \right\rfloor$. Załóżmy teraz, że $\left\lfloor f(\lfloor x \rfloor) \right\rfloor < \left\lfloor f(x) \right\rfloor$. Stąd oczywiście $f(\lfloor x \rfloor) < \left\lfloor f(x) \right\rfloor \leq f(x)$, a więc istnieje liczba $\lfloor x \rfloor < y \leq x$ taka, że $f(y) = \left\lfloor f(x) \right\rfloor$. Z własności funkcji $f()$ wynika, że y jest liczbą całkowitą (bo $f(y)$ jest liczbą całkowitą) a nie ma żadnej liczby całkowitej pomiędzy $\lfloor x \rfloor$ a x. □

Wniosek 3.1.1. *Dla dowolnych* $n, m \in \mathcal{N}, x \in \Re$, *bezpośrednio z twierdzenia 3.1.3 otrzymujemy:*

$$\begin{aligned} \left\lfloor \frac{x+m}{n} \right\rfloor &= \left\lfloor \frac{\lfloor x \rfloor + m}{n} \right\rfloor, \\ \left\lceil \frac{x+m}{n} \right\rceil &= \left\lceil \frac{\lceil x \rceil + m}{n} \right\rceil. \end{aligned}$$ □

Problem 3.1.2. *Czy prawdziwe są równości*

$$\begin{aligned} \left\lfloor \sqrt{\lceil x \rceil} \right\rfloor &= \left\lfloor \sqrt{x} \right\rfloor, \\ \left\lceil \sqrt{\lfloor x \rfloor} \right\rceil &= \left\lceil \sqrt{x} \right\rceil ? \end{aligned}$$

□

Oznaczmy przez

$[\alpha, \beta]$	przedział domknięty $\alpha \le x \le \beta$,
$[\alpha, \beta)$	przedział $\alpha \le x < \beta$,
$(\alpha, \beta]$	przedział $\alpha < x \le \beta$,
(α, β)	przedział otwarty $\alpha < x < \beta$.

Twierdzenie 3.1.4. *Ilość liczb całkowitych zawartych w każdym z tych przedziałów przedstawiamy w Tabeli 3.1*

Tabela 3.1. Ilość liczb całkowitych w różnego typu przedziałach.

Przedział	Ilość liczb całkowitych zawartych w tym przedziale	Założenia
$[\alpha, \beta]$	$\lfloor \beta \rfloor - \lceil \alpha \rceil + 1$	$\alpha \le \beta$,
$[\alpha, \beta)$	$\lceil \beta \rceil - \lceil \alpha \rceil$	$\alpha \le \beta$,
$(\alpha, \beta]$	$\lfloor \beta \rfloor - \lfloor \alpha \rfloor$	$\alpha \le \beta$,
(α, β)	$\lceil \beta \rceil - \lfloor \alpha \rfloor - 1$	$\alpha < \beta$,

Dowód. Jeśli $\alpha \le n \le \beta$ to $\lceil \alpha \rceil \le n \le \lfloor \beta \rfloor$ a liczb naturalnych spełniających ten warunek jest dokładnie $\lfloor \beta \rfloor - \lceil \alpha \rceil + 1$. Podobnie jeśli $\alpha \le n < \beta$ to $\lceil \alpha \rceil \le n < \lceil \beta \rceil$ a liczb naturalnych spełniających ten warunek jest dokładnie $\lceil \beta \rceil - \lceil \alpha \rceil$. Pozostałe wzory dowodzimy podobnie. □

Problem 3.1.3. *Ile jest wielokrotności liczby x w przedziałach: $[\alpha, \beta]$, $[\alpha, \beta)$, $(\alpha, \beta]$, (α, β)?* □

Przykład 3.1.1. *Obliczmy sumę $\sum_{0 \le k < n} \lfloor \frac{k}{2} \rfloor$.*

Mamy

$$\begin{aligned} \sum_{0 \le k < n} \lfloor \frac{k}{2} \rfloor &= \sum_{k,j \ge 0} j[j = \lfloor \frac{k}{2} \rfloor][k < n] \\ &= \sum_{k,j \ge 0} j[j \le \frac{k}{2} < j + 1][k < n] \\ &= \sum_{k,j \ge 0} j[2j \le k < 2j + 2][k < n] \end{aligned}$$

$$
\begin{aligned}
&= \sum_{j\geq 0}\sum_{k\geq 0} j[2j \leq k < 2j+2][k<n] \\
&= \sum_{j\geq 0} j \sum_{k\geq 0}[2j \leq k < 2j+2 \leq n] + \sum_{j\geq 0} j \sum_{k\geq 0}[2j \leq k < n < 2j+2] \\
&= \sum_{j\geq 0} j(2j+2-2j)[2j+2 \leq n] + \sum_{j\geq 0} j(n-2j)[2j < n < 2j+2] \\
&= \sum_{j\geq 0} 2j[2j \leq n-2] + \sum_{j\geq 0} j(n-2j)[2j+1 = n = 2j+1] \\
&= 2\sum_{0\leq j\leq \lfloor \frac{n}{2}\rfloor -1} j + \frac{n-1}{2}[n-1 \ \ \textit{parzyste}] \\
&= (\lfloor \frac{n}{2}\rfloor -1)(\lfloor \frac{n}{2}\rfloor) + \frac{n-1}{2}[n-1 \ \ \textit{parzyste}]. \qquad \square
\end{aligned}
$$

Problem 3.1.4. *Oblicz następujące sumy* $\sum_{0\leq k<n}\lceil \frac{k}{2}\rceil, \sum_{0\leq k<n}\lfloor \frac{k}{3}\rfloor, \sum_{0\leq k<n}\lceil \frac{k}{3}\rceil$. $\square$

3.2. Widmo liczby

Definicja 3.2.1. *Widmem dowolnej liczby rzeczywistej* α *nazywamy ciąg* $Spec(\alpha) = \{\lfloor \alpha\rfloor, \lfloor 2\alpha\rfloor, \lfloor 3\alpha\rfloor, \lfloor 4\alpha\rfloor, ...\}$. $\square$

Na przykład:

$$
\begin{aligned}
Spec(\frac{1}{2}) &= \{0,1,1,2,2,3,3,...\}, \\
Spec(\frac{1}{3}) &= \{0,0,1,1,1,2,2,2,3,3,3,...\}, \\
Spec(-2.3) &= \{-3,-5,-7,-10,-12,-14,-17,...\}, \\
Spec(\pi) &= \{3,6,9,12,15,18,21,25,28,31,...\}.
\end{aligned}
$$

Łatwo sprawdzić, że ciąg ten jest niemalejący dla dodatnich liczb α, nierosnący dla ujemnych i stały równy 0 dla $\alpha = 0$.

Twierdzenie 3.2.1. *Dla dowolnych liczb rzeczywistych* α *i* β*:*

$$\alpha = \beta \Longleftrightarrow Spec(\alpha) = Spec(\beta).$$

Dowód. Implikacja $\alpha = \beta \Rightarrow Spec(\alpha) = Spec(\beta)$ jest oczywista. Jeśli $\alpha \neq \beta$ to można z symetrii założyć, że $\alpha < \beta$, a wtedy istnieje taka liczba

naturalna m, że $m(\beta-\alpha)\geq 1$ (np. $m=\lceil\frac{1}{\beta-\alpha}\rceil$) stąd $m\beta\geq m\alpha+1$ czyli $\lfloor m\beta\rfloor>\lfloor m\alpha\rfloor$ a stąd $Spec(\alpha)\neq Spec(\beta)$. □

Reprezentacja liczby jej widmem jest więc reprezentacją jednoznaczną. Przez $Spec_k(\alpha)$ oznaczać będziemy k-ty element widma α, to znaczy $\lfloor k\alpha\rfloor$.

Dla dowolnej **dodatniej** liczby α oznaczmy przez $N(\alpha,n)$ ilość elementów w widmie $Spec(\alpha)$ nie większych niż n. Wtedy

$$\begin{aligned}
N(\alpha,n) &= \sum_{k>0}[\lfloor k\alpha\rfloor\leq n]\\
&= \sum_{k>0}[\lfloor k\alpha\rfloor<n+1]\\
&= \sum_{k>0}[k\alpha<n]\\
&= \sum_{k}[0<k<(n+1)/\alpha]\\
&= \lceil(n+1)/\alpha\rceil-1.
\end{aligned}$$

Twierdzenie 3.2.2. *Niech p i q będą dwoma liczbami niewymiernymi takimi, że $\frac{1}{p}+\frac{1}{q}=1$. Wtedy $Spec(p)\cup Spec(q)=\mathcal{N}$, $Spec(p)\cap Spec(q)=\emptyset$.*

Dowód. Napierw wykażemy że dla każdej liczby naturalnej n zachodzi $N(p,n)+N(q,n)=n$ (zwróćmy uwagę, że równość ta oznacza iż liczba n występuje dokładnie raz w jednym z widm). Mamy

$$\begin{aligned}
& N(p,n)+N(q,n)=\lceil\frac{n+1}{p}\rceil-1+\lceil\frac{n+1}{q}\rceil-1=n\\
\iff\ & \lfloor\frac{n+1}{p}\rfloor+\lfloor\frac{n+1}{q}\rfloor=n,\\
\iff\ & \frac{n+1}{p}-\{\frac{n+1}{p}\}+\frac{n+1}{q}-\{\frac{n+1}{q}\}=n,\\
\iff\ & (n+1)(\frac{1}{p}+\frac{1}{q})-\{\frac{n+1}{p}\}-\{\frac{n+1}{q}\}=n,\\
\iff\ & \{\frac{n+1}{p}\}+\{\frac{n+1}{q}\}=1,
\end{aligned}$$

co jest prawdziwe bo $\frac{n+1}{p}+\frac{n+1}{q}=n+1$. □

W szczególności za liczby p i q można wybrać $\sqrt{2}$ i $2+\sqrt{2}$ (Sprawdź).

3.3. Rekurencje z funkcjami podłoga i sufit

Przykład 3.3.1. *Niech $a[1:n]$ będzie tablicą liczb, które chcemy posortować a $b[1:n]$ pomocniczą tablicą. Rozważmy następujący algorytm sortowania:*

Algorytm 3.1. Sortowanie przez podział i złączanie

```
procedure sort(p,k: integer);
    p1,i,j,s : integer;
    if k>p then
          {
              p1:=⌊ (k+p)/2 ⌋;
              sort(p,p1);
              sort(p1+1,k);
              i:=p; j:=p1+1; s:=p;
              while (i<=p1) and (j<=k) do
                 {
                    if a[i]<a[j] then
                        { b[s]:=a[i]; i:=i+1; }
                    else
                        { b[s]:=a[j]; j:=j+1; }
                    s:=s+1;
                 }
              if i=p1+1 then
                 {
                    for i:=j to k
                      { b[s]:=a[i]; s:=s+1; }
                 }
              else
                 {
                    for j:=i to p1
                       { b[s]:=a[j]; s:=s+1; }
                 }
              for i:=p to k
                a[i]:=b[i];
          }
sort(1,n);
```

Wyznacz liczbę porównań elementów tablicy a w najgorszym możliwym przypadku. Powyższą metodę sortowania możemy opisać następująco: dzielimy ciąg na dwie równe (gdy n nieparzyste "mniej więcej" równe) części, sortujemy te części osobno i potem łączymy je w jeden posortowany już ciąg. Dzielenie następuje tak długo, dopóki nie otrzymamy ciąg długości 1. Podział na "mniej więcej" równe posortowane części odpowiada przedstawieniu liczby n w następujący sposób:

$$n = \lfloor \frac{n}{2} \rfloor + \lceil \frac{n}{2} \rceil,$$

a więc uzyskujemy rekurencję:

$$\begin{cases} T(0) &= 0, \\ T(n) &= T(\lfloor \frac{n}{2} \rfloor) + T(\lceil \frac{n}{2} \rceil) + (n-1), \quad dla\ n \geq 1. \end{cases}$$

Jak rozwiązać tę rekurencję? Udowodnimy najpierw pomocniczą tożsamość:

$$\begin{aligned} \lceil \log_2(2j) \rceil &= \lceil \log_2(j) \rceil + 1, \;\; j \geq 1, \\ \lceil \log_2(2j-1) \rceil &= \lceil \log_2(j) \rceil + [j > 1], \;\; j \geq 1. \end{aligned}$$

Pierwsza równość wynika z faktu, że $\lceil \log_2(2j) \rceil = \lceil \log_2(j) + 1 \rceil$*, natomiast dla drugiej mamy*

$$\begin{aligned} & \lceil \log_2(2j-1) \rceil = k \\ \iff \;& k - 1 < \log_2(2j-1) \leq k \\ \iff \;& 2^{k-1} < 2j - 1 \leq 2^k \\ \iff \;& 2^{k-2} + \frac{1}{2} < j \leq 2^{k-1} + \frac{1}{2} \\ \iff \;& 2^{k-2} < j \leq 2^{k+[j>1]}, \end{aligned}$$

bowiem jedynie dla $k = 1$ *w przedziałach typu* $(2^{k-1}, 2^{k-1} + \frac{1}{2}]$ *znajduje się liczba całkowita.*

Następnie rozważmy sumę $\sum_{k=1}^{n} \lceil \log_2(k) \rceil$*. Ponieważ*

$$\begin{aligned} \sum_{k=1}^{n} \lceil \log_2(k) \rceil &= \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} \lceil \log_2(2k) \rceil + \sum_{k=1}^{\lceil \frac{n}{2} \rceil} \lceil \log_2(2k-1) \rceil \\ &= \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} \lceil \log_2(k) \rceil + \lfloor \frac{n}{2} \rfloor + \sum_{k=1}^{\lceil \frac{n}{2} \rceil} \lceil \log_2(k) \rceil + \lceil \frac{n}{2} \rceil - 1 \\ &= \sum_{k=1}^{\lfloor \frac{n}{2} \rfloor} \lceil \log_2(k) \rceil + \sum_{k=1}^{\lceil \frac{n}{2} \rceil} \lceil \log_2(k) \rceil + n - 1, \end{aligned}$$

więc

$$T(n) = \sum_{k=1}^{n} \lceil \log_2(k) \rceil. \qquad \square$$

Ta sama suma pojawia się w innym algorytmie sortowania - algorytmie wstawiania połówkowego (algorytm 7.2). Algorytm ten zostanie omówiony szczegółowo w podrozdziale 7.3.1.

Przykład 3.3.2. Rekurencja Flawiusza. *Rozważmy teraz ponownie rekurencję Flawiusza. Rekurencję Flawiusza* $J_2()$ *możemy zapisać zwartym wzorem jako:*

$$\begin{cases} J_2(1) &= 1, \\ J_2(n) &= 2J_2(\lfloor \frac{n}{2} \rfloor) - (-1)^n, \;\; dla \;\; n \geq 2, \end{cases}$$

natomiast "oryginalną" rekurencję Flawiusza można opisać

$$\begin{cases} J_3(1) &= 1, \\ J_3(n) &= \lceil \frac{3}{2} J_3(\lfloor \frac{2n}{3} \rfloor) + a_n \rceil \bmod n + 1, \quad dla \;\; n \geq 2, \end{cases}$$

gdzie

$$a_n = \begin{cases} -2 & jeśli \;\; n \bmod 3 = 0, \\ 1 & jeśli \;\; n \bmod 3 = 1, \\ -\frac{1}{2} & jeśli \;\; n \bmod 3 = 2, \end{cases}$$

a operator dwuargumentowy n mod m oznacza resztę z dzielenia n przez m (więcej o tym operatorze w następnym podrozdziale). Nie jest to równanie, które możnaby rozwiązać jakimiś znanymi metodami, ale zupełnie inne podejście do problemu Flawiusza może być efektywniejsze. Załóżmy, że po eliminacjach nadamy nowe (dalsze) numery pozostałym osobom. Na przykład, dla n = 12 osób mamy

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
13	*14*		*15*	*16*		*17*	*18*		*19*	*20*	
21	*22*			*23*		*24*			*25*	*26*	
	27			*28*					*29*	*30*	
				31					*32*		
				33					*34*		
									35		
									36		

Zauważmy, że k-ta w kolejności zabita osoba teraz w chwili śmierci ma numer 3k a poszukiwana $J_3(n)$-ta osoba ma numer 3n. Jak łatwo widać osoba o numerze $N = n + 2k + 1$ lub $N = n + 2k + 2$ miała numer $3k + 1$ lub $3k + 2$, odpowiednio, a stąd "stary" numer jest równy $\lfloor \frac{N-n-1}{2} \rfloor + N - n$ co prowadzi do następującego algorytmu obliczania $J_3(n)$:

Algorytm 3.2. Obliczanie $J_3(n)$

```
N:=3n;
while N>n do
    N:= ⌊(N-n-1)/2⌋+N-n;
J3(n):=N;
```

Otrzymaliśmy więc algorytm obliczenia $J_3(n)$ ale dalej nie mamy tutaj ani zwartego wzoru, ani nawet rekurencji. Jeśli jednak zrobimy podstawienie $D = 3n + 1 - N$ to

$$D := 3n + 1 - \left(\left\lfloor \frac{(3n + 1 - D) - n - 1}{2} \right\rfloor + (3n + 1 - D) - n \right)$$

$$\begin{aligned} &= n + D - \Big\lfloor \frac{2n - D}{2} \Big\rfloor = d - \Big\lfloor \frac{-D}{2} \Big\rfloor = d + \Big\lceil \frac{D}{2} \Big\rceil \\ &= \Big\lceil \frac{3}{2} D \Big\rceil, \end{aligned}$$

a stąd otrzymujemy algorytm

Algorytm 3.3. Obliczanie $J_3(n)$ - drugi sposób

```
D:=1;
while D<=2n do
      D:= ⌈3/2 D⌉;
J3(n):=3n+1−D;
```

Zdefiniujmy ciąg rekurencyjny

$$\begin{cases} D_0^{(q)} &= 1, \\ D_n^{(q)} &= \lceil \frac{q}{q-1} D_{n-1}^{(q)} \rceil, \quad dla \ n \geq 1, \end{cases}$$

wtedy

$$J_q(n) = qn + 1 - D_k^{(q)},$$

gdzie $k = \inf\{j \geq 1 : D_j^{(q)} > (q-1)n\}$.

W poniższej tabeli podamy liczby $D_k^{(q)}, 1 \leq k \leq 30, q = 3, 4, 5$:

Tabela 3.2. Liczby $D_k^{(q)}, q = 3, 4, 5$.

k	$D_k^{(3)}$	$D_k^{(4)}$	$D_k^{(5)}$	k	$D_k^{(3)}$	$D_k^{(4)}$	$D_k^{(5)}$
1	1	1	1	16	710	154	75
2	2	2	2	17	1065	206	94
3	3	3	3	18	1598	275	118
4	5	4	4	19	2397	367	148
5	8	6	5	20	3596	490	185
6	12	8	7	21	5394	654	232
7	18	11	9	22	8091	872	290
8	27	15	12	23	12137	1163	363
9	41	20	15	24	18206	1551	454
10	62	27	19	25	27309	2068	568
11	93	36	24	26	40964	2758	710
12	140	48	30	27	61446	3678	888
13	210	64	38	28	92169	4904	1110
14	315	86	48	29	138254	6539	1388
15	473	115	60	30	207381	8719	1735

Obliczmy teraz $J_3(41)$. *Z tabelki widzimy, że* $D_{11}^{(3)} > 2 \times 41 = 82$ *a więc* $k = 11$ *a* $J_3(41) = 3 \times 41 + 1 - 93 = 31$. *Należy więc stanąć na* 31 *pozycji. Widzimy więc jak dobrym matematykiem był Flawiusz (matematyka czasem ratuje ludziom życie).*

Sprawdźmy jeszcze przedstawioną teorię dla znanego przypadku $q = 2$. *Wtedy* $D_n^{(2)} = 2^n$ *i* $k = \inf\{j \geq 1 : D_j^{(2)} > n\} = m(n) + 1 = \lfloor \log_2(n) \rfloor + 1$ *a więc*

$$J_2(n) = n + 1 - D_k^{(2)} = 2n + 1 - 2^{m+1} = 2(n - 2^m) + 1 = 2l + 1,$$

co jest zgodne z Twierdzeniem 1.2.1. □

3.4. Działanie dwuargumentowe *mod*

W poprzednim podrozdziale pojawił się operator dwuargumentowy *mod*. Zajmiemy się tutaj bliżej własnościami tego operatora. Dla dowolnych liczb naturalnych n i m $(n > m)$, liczba m mieści się $\lfloor \frac{n}{m} \rfloor$ razy w liczbie n, natomiast $n \ mod \ m$ jest resztą z dzielenia n przez m, a więc każdą liczbę naturalną n możemy przedstawić w postaci

$$n = \lfloor \frac{n}{m} \rfloor m + n \mod m. \tag{3.4.1}$$

Chcemy rozszerzyć tę własność na wszystkie liczby rzeczywiste, kładąc dla $x, y \in \Re, y \neq 0$

$$x \mod y = x - y \lfloor \frac{x}{y} \rfloor.$$

Dla $y = 0$ dookreślamy

$$x \ mod \ 0 = x.$$

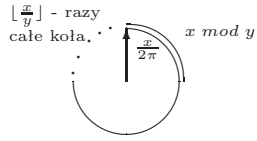
Definicja 3.4.1. *Operatorem dwuargumentowym* $x \ mod \ y$ *określonym dla* $x, y \in \Re$ *nazywamy*

$$x \ mod \ y = \begin{cases} x - y \lfloor \frac{x}{y} \rfloor, & dla \ \ y \neq 0, \\ x, & dla \ \ y = 0. \end{cases}$$ □

Liczbę za symbolem y nazywamy modulnikiem. Możemy spytać się jaka jest interpretacja geometryczna tej operacji? Wyobraźmy sobie okrąg o obwodzie długości y. Jeżeli przejdziemy wzdłuż tego okręgu drogę długości

x to wykonamy $\lfloor \frac{x}{y} \rfloor$ pełne obroty okręgu a skończymy w miejscu odległym od początku o $x \ mod \ y$.

Rysunek 3.3. Ilustracja $x \ mod \ y$ oraz $\lfloor \frac{x}{y} \rfloor$.



Twierdzenie 3.4.1. *Dla dowolnych x, y, c mamy:*

$$\begin{array}{lrcl} \text{(i)} & 0 \leq x \ mod \ y & < & y, \ dla \ y > 0, \\ \text{(ii)} & 0 \geq x \ mod \ y & > & y, \ dla \ y < 0, \\ \text{(iii)} & c(x \ mod \ y) & = & (cx) \ mod \ (cy). \end{array}$$

Dowód (iii)

$$c(x \ mod \ y) = c(x - y\lfloor \frac{x}{y} \rfloor) = cx - cy\lfloor \frac{cx}{cy} \rfloor = (cx) \ mod \ (cy). \qquad \square$$

Problem 3.4.1. *Podziel n osób na m możliwie równolicznych grup. Na przykład niech $n = 10$ dzielimy na $m = 4$ grupy. Otrzymujemy dwie grupy po $\lfloor \frac{10}{4} \rfloor = 2$ elementów oraz dwie grupy po $\lceil \frac{10}{4} \rceil = 3$ elementów, czyli*

$$10 = \lfloor \frac{10}{4} \rfloor + \lfloor \frac{10}{4} \rfloor + \lceil \frac{10}{4} \rceil + \lceil \frac{10}{4} \rceil.$$

Ogólnie

$$n = \underbrace{\lfloor \frac{n}{m} \rfloor + \lfloor \frac{n}{m} \rfloor + ... + \lfloor \frac{n}{m} \rfloor}_{n \ mod \ m} + \underbrace{\lceil \frac{n}{m} \rceil + \lceil \frac{n}{m} \rceil + ... + \lceil \frac{n}{m} \rceil}_{m-n \ mod \ m},$$

czyli

$$n = \lfloor \frac{n}{m} \rfloor + \lfloor \frac{n+1}{m} \rfloor + ... + \lfloor \frac{n+m-1}{m} \rfloor,$$

lub

$$n = \lceil \frac{n}{m} \rceil + \lceil \frac{n-1}{m} \rceil + ... + \lceil \frac{n-m+1}{m} \rceil. \qquad \square$$

Niech teraz $n = \lceil mx \rceil$. Wtedy z Wniosku 3.1.1

$$\lfloor mx \rfloor = \lfloor x + \frac{1}{m} \rfloor + \lfloor x + \frac{2}{m} \rfloor + ... + \lfloor x + \frac{m-1}{m} \rfloor. \tag{3.4.2}$$

Analogicznie możemy otrzymać

$$\lceil mx \rceil = \lceil x - \frac{1}{m} \rceil + \lceil x - \frac{2}{m} \rceil + ... + \lceil x - \frac{m-1}{m} \rceil. \tag{3.4.3}$$

3.5. Generatory liczb pseudolosowych

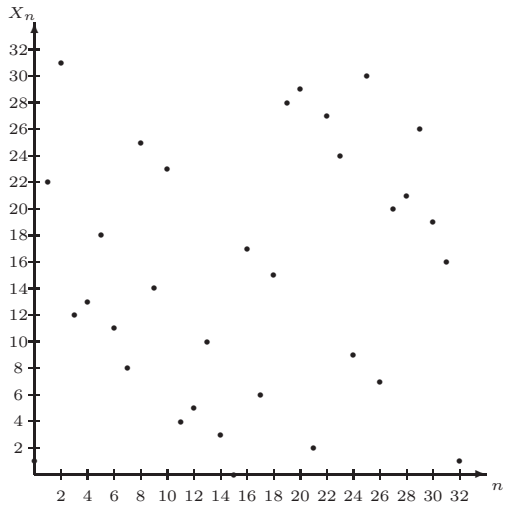
Ważnym problemem, w zastosowaniach komputerowych, jest wygenerowanie ciągu losowego. Komputer jest urządzeniem w 100% deterministycznym, a podłączenie jakiejś przystawki, gdzie byłby generowany proces losowy, chociaż możliwe (oparte na szumie termicznym lub promieniowaniu kosmicznym) jest bardzo drogie. Zastosowanie w komputerze losowych liczb jest ogromne: różne symulacje, obliczanie całek czy też rozwiązywanie równań, których w inny sposób nie można obliczyć, gry, projektowanie itp. Przy braku rzeczywistej losowej przystawki jedyną metodą generowania ciągu liczb w komputerze jest ciąg postaci $X_n = f(X_{n-1}, ..., X_{n-k}), n \geq k+1$, przy pewnych warunkach początkowych. Niestety tak wygenerowany ciąg jest **zawsze (!)** okresowy, a zgodnie z naszą intuicją, losowy ciąg powinien być nieokresowy. Podobnie, ze względu na skończoność pamięci komputera ciąg "losowy" w komputerze składa się wyłacznie z liczb wymiernych, a prawdopodobieństwo wylosowania liczby wymiernej z przedziału $[0, 1]$ wynosi 0(liczby niewymiernej wynosi 1). Dlatego ciągi "losowe" tworzone w komputerze określamy jako psudolosowe a nie losowe (z wyjątkiem wspomnianych na początku losowych przystawek). O jakości ciągu pseudolosowego świadczy więc długość okresu, szybkość obliczeń i spełnianie pewnej grupy testów statystycznych. Chyba jednym z częściej używanych takich ciągów jest generator liniowy kongruencyjny lub generator liniowy Lehmera (LCG):

$$LCG(M, a, b, \alpha): \begin{cases} X_o &= \alpha, \\ X_n &= (aX_{n-1} + b) \; mod \; M, \quad n > 1, \end{cases} \tag{3.5.4}$$

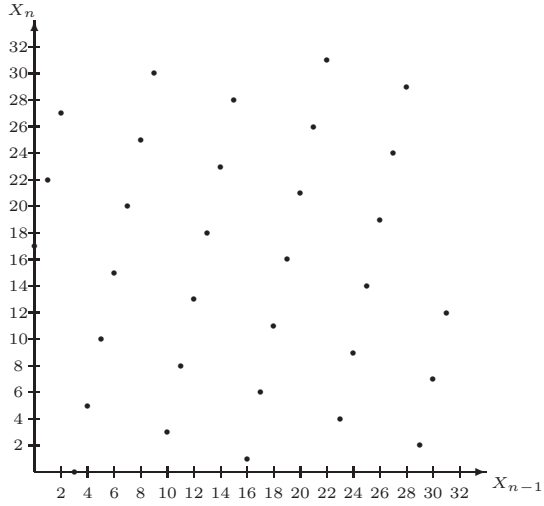
gdzie możliwie dobrym wyborem jest $M = 2^m, c = 2k+1, a = 4l+1, k, l, m \in \mathcal{N}$, wtedy ciąg ten ma okres 2^m, największy jaki może przyjmować. W ANSI-C jako funkcja $rand()$ stosowany jest $LCG(2^{31}, 1103515245, 12345, 12345)$ z, jak widać, dość prostymi wartościami b i α. Inny generator liniowy kongruencyjny $LCG(2^{31} - 1, 6300360016, 0, 0)$ jest wykorzystywany w FORTRANIE w funkcji $RAN()$ a jeszcze inne wykorzystywno w CUPL, BCPL, Turbo C++, Turbo Pascalu i Mapple. Generatory LCG, używane w symulacjach i metodach Monte Carlo (np., numeryczne obliczanie całek), są jednak całkowicie nieprzydatne w kryptografii, kolejne wartości przyrostów generatora układają się bowiem na liniach prostych ($y = ax + b$) o tym samym współczynniku nachylenia. Po odgadnięciu pierwszej wartości generetora pozostałe wartości jest więc już zbyt łatwo złamać (wzór (3.5.4)

jest za prosty). Na rysunkach 3.4 i 3.5 przedstawiliśmy wykres generatora $LCG(32, 5, 17, 0)$ i przyrosty tego generatora.

Rysunek 3.4. Generator $LCG(32, 5, 17, 0)$. Wykres zależności X_n od n.



Rysunek 3.5. Generator $LCG(32, 5, 17, 0)$. Wykres zależności X_n od X_{n-1}.



Inne, często używane generatory to generator multyplikatywny (we wzorze (3.5.4) kładziemy $b = 0$), generatory odwrotne, generatory oparte na rejestrach przesuwnych czy też generatory Fibonacciego. Wszystkie one oparte są na operatorze $x \; mod \; y$.

Przeanalizujmy jeszcze wzór (3.5.4). Czy można podać nierekurencyjny wzór na X_n? Podstawmy $\alpha' = \alpha/M, a' = a, b' = b/M$, oraz $Y_n = \frac{X_n}{M}, n \geq 0$. Wtedy (z Twierdzenia 3.4.1. (iii)) (3.5.4) jest równoważne

$$LCG(1, a', b', \alpha') : \left\{ \begin{array}{lcl} Y_o & = & \alpha', \\ Y_n & = & (a'Y_{n-1} + b') \; mod \; 1 = \{a'Y_{n-1} + b'\}, \quad n > 1, \end{array} \right.$$

gdzie $X_n = Y_n M, n \geq 0$. Ponieważ a' jest liczbą całkowitą, więc

$$\begin{array}{rcl} Y_n & = & (a'(a'Y_{n-2} + b' \; mod \; 1) + b') \; mod \; 1 \\ & = & (a'(a'Y_{n-2} + b') - a'\lfloor a'Y_{n-2} + b' \rfloor + b') \; mod \; 1 \\ & = & (a'^2 Y_{n-2} + a'b' + b') \; mod \; 1. \end{array}$$

Rozumując analogicznie dalej, widzimy, że jeśli Z_n jest rozwiązaniem rekurencji

$$\left\{ \begin{array}{lcl} Z_o & = & \alpha', \\ Z_n & = & a'Z_{n-1} + b', \quad n > 1, \end{array} \right. \tag{3.5.5}$$

to

$$Y_n = Z_n \; mod \; 1 = \{Z_n\}, n \geq 0.$$

Rekurencję (3.5.5) możemy rozwiązać metodą czynnika sumacyjnego, kładąc we wzorze (2.2.1) $a_0 = 1, c_0 = \alpha', a_n = 1, b_n = a', n \geq 1, c_n = b'$ mamy $s_n = 1/a'^n, n \geq 1$, ponadto z (2.2.3) otrzymujemy

$$\begin{array}{rcl} Z_n & = & \frac{1}{\frac{1}{a'^n}}(\frac{1}{a'}a'\alpha' + \sum_{k=1}^{n} \frac{1}{a'^k} b') \\ & = & \alpha' a'^n + \frac{a'^n - 1}{a' - 1} b' \\ & = & \frac{\alpha a^n + \frac{a^n - 1}{a-1} b}{M}, \end{array}$$

a zatem z Twierdzenia 3.4.1 (iii)

$$LCG(M, a, b, \alpha) : \left\{ \begin{array}{lcl} X_o & = & \alpha, \\ X_n & = & (a^n \alpha + \frac{a^n - 1}{a-1} b) \; mod \; M. \quad n > 1. \end{array} \right. \qquad \square$$

3.6. Zapisy liczb naturalnych w różnych podstawach liczenia

Nasz układ liczenia ma podstawę 10 ale w komputerze informacja jest zapisywana w systemie o podstawie 2 (binarnie) lub o podstawie 16 (heksadecymalnie). Sposób zapisu liczb wyznacza też w jaki sposób wykonujemy

pisemnie obliczenia, inaczej bowiem wtedy wyglądają tabliczki mnożenia i dodawania. O binarnym układzie liczenia pisaliśmy w Uwadze 1.2.4, tutaj omówimy problem zamiany zapisów liczb z układów liczenia o różnych podstawach.

Dawniej, w niektórych krajach, używano innych systemów liczenia niż dziesiętny, na przykład Babilończycy używali systemu 60-tnego, stąd do dziś godzina ma 60 minut, minuta 60 sekund itp. Również pozostałością innych niż 10-tne sposoby liczenia są takie pojęcia jak "kopa" czy "tuzin". Liczby w układzie o podstawie $p = 2, 3, \ldots$ wykorzystują p cyfr, stąd liczbę w układzie dwójkowym zapisujemy korzystając z cyfr $\{0, 1\}$, siódemkowym $\{0, 1, 2, 3, 4, 5, 6\}$ a szesnastkowym $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$, gdzie $A = 10, B = 11, C = 12, D = 13, E = 14, F = 15$.

Definicja 3.6.1. *Mówimy, że liczba rzeczywista x ma zapis w układzie o podstawie p równy $\pm(c_n c_{n-1} ... c_1 c_0, c_{-1} c_{-2} ... c_{-k})_p$ jeżeli*

$$x = sign(x)(c_n c_{n-1} ... c_1 c_0, c_{-1} c_{-2} ... c_{-k})_p = sign(x) \sum_{i=-k}^{n} c_i p^i. \qquad (3.6.6)$$

przy czym dopuszczamy $k = \infty$. □

Podobnie jak w dziesiętnym układzie liczenia w dowolnym układzie liczenia liczby wymierne mają zawsze skończony okres a liczby niewymierne nie. Na przykład

$$\begin{aligned}
\frac{2}{3} &= (0.(6))_{10} = (0.(10))_2 = (0, (31))_5 = (0.(4))_7 = (0.(52))_8 = (0.(A))_{16} \\
-3725,125 &= -(3725, 125)_{10} = -(111010001101, 001)_2 \\
&= -(104400, (03))_5 = -(13601, (06))_7 = -(7215, 1)_8 = -(21A5, 16)_{16} \\
203,22 &= (203, 22)_{10} = (11001011, 0(01110000101000111101))_2 = (1303, 10(2))_5 \\
&= (410, (1353))_7 = (313, 1(60507534121727024365))_8 = (CB, 3852)_{16}
\end{aligned}$$

W jaki sposób zamieniać liczby zapisane w jednej podstawie na liczby zapisane w innej podstawie. Zajmijmy się najpierw liczbami naturalnymi. Jeśli $x = \sum_{j=0}^{n} c_j p^j$, oraz $0 \leq c_j < p, j \geq 0$ to $\lfloor \frac{x}{p} \rfloor = \sum_{j=1}^{n} c_j p^{j-1}$ oraz $x \ mod \ p = c_0$. Stąd, aby zapisać liczbę 203 w układzie dwójkowym obliczamy część całkowitą ilorazu $\lfloor \frac{203}{2} \rfloor = 101$ oraz $203 \ mod \ 2 = 1 = c_0$, dalej znowu $\lfloor \frac{101}{2} \rfloor = 50$ i $101 \ mod \ 2 = 1 = c_1$. Możemy ten proces opisać w tabelce

203	101	50	25	12	6	3	1
1	1	0	1	0	0	1	1

a stąd $203_{10} = (11001011)_2$, (zauważ, że cyfry piszemy w odwrotnej kolejności).

Niech teraz y będzie ułamkiem o podstawie p, to znaczy $y = \sum_{i=-k}^{-1} c_i p^i$ oraz $0 \leq c_i < p, i \leq -1$, wtedy $\lfloor yp \rfloor = c_{-1}$ a $yp - \lfloor yp \rfloor = \sum_{i=-k+1}^{-1} c_i p^{i+1}$. W przypadku $y = 0,22$ mamy $2y = 0,44$ i $c_{-1} = 0$, dalej $2 \cdot 0,44 = 0,88$ i $c_{-2} = 0$, $2 \cdot 0,88 = 1,76$ i $c_{-3} = 1$ teraz $2 \cdot 0,76 = 1,52$ i $c_{-4} = 1$ itd. Możemy ten proces przedstawić w formie tabelki

0,22	0,44	0,88	0,76	0,52	0,04	0,08	0,16	0,32	0,64	0,28	0,56
0	0	1	1	1	0	0	0	0	1	0	1

0,12	0,24	0,48	0,96	0,92	0,84	0,68	0,36	0,72	0,44	...	
0	0	0	1	1	1	1	0	1	0	...	

a stąd $0,22_{10} = (0,0(01110000101000111101))_2$. Znak liczby przepisujemy bez zmian.

Rozważania te prowadzą do następującego algorytmu

Algorytm konstrukcji rozwinięcia liczby n o podstawie b

Dane: Liczba x.

Wynik: Napis $\pm(c_n, c_{n-1}, c_{n-2}, ..., c_0, c_1 c_2 c_3 ... c_k)_p$.

Uwaga: Należy pamiętać, że gdy liczba x ma rozwinięcie okresowe lub nieskończone części ułamkowej to obliczanie tej części nam się zapętli, dobrze by było dorzucić licznik ograniczający ilość tych cykli!

Uwaga: W rzeczywistych obliczeniach pojawiają się nam dość szybko błędy zaokrągleń, (błędy wynikające z utraty cyfr znaczących ułamków reprezentujących liczby zmiennopozycyjne, powodowane koniecznością obcięcia wyników obliczeń w związku ze skończoną reprezentacją komputerową liczb), stąd dobrze by było wykonać obliczenia możliwie dużej precyzji!

Algorytm 3.4. Konstrukcja rozwinięcia liczby n o podstawie b

```
if x<0 then znakx="-" else znakx="+";
x1=Abs(x);  intx=⌊x1⌋; decx=x1-intx;

c="";
{Część całkowita}
while intx≠0
    {
        q=intx mod p;
        if q>9 then
            c := Chr(q+22) & c      {Dodajemy literę A=10, B=11 ...
                }
        else
            c := Str(q) & c;        {Dodajemy cyfrę}
        intx := ⌊intx/p⌋;
```

```
      }
{Część ułamkowa}
c=c & ",";
while decx≠0
    {
        q=⌊p*decx ⌋
        if q>9 then
            c := c & Chr(q+22)      {Dodajemy literę A=10, B=11... }
        else
            c := c & Str(q);        {Dodajemy cyfrę}
        decx := p*decx-⌊p*decx⌋;
    }
{Dodaj znak }
c := znakx & c;
{Wynikiem jest łańcuch c}
```

Przy obliczaniu liczby (w układzie dziesiętnym) na podstawie jej rozwinięcia korzystamy ze wzoru (3.6.6). Dobrze jest wtedy wykorzystać tak zwany schemat Hornera, stosowany do obliczeń wartości wielomianu w jakimś punkcie. Zwróćmy, uwagę, że dla zadanej tablicy $c[i], i = -k, \ldots n$, algorytm

```
s:=c[0]; p1:=p;
for i:=1 to n
   s:=s+c[i]*p1;
   p1:=p1*p;
p1:=1/p;
for i:=1 to k
   s:=s+c[-i]*p1;
   p1:=p1/p;
```

wymaga $2 * (n + k) + 1$ operacji mnożenia/dzielenia i $n + k$ dodawania, podczas gdy

```
s:=c[n];
for i:=n-1 to 1 step -1
   s:=s*p+c[i];
s1:=c[-k];
for i:=-k+1 to -1
   s1:=s1/p+c[i];
s:=s*p+s1/p+c[0];
```

tylko $n+k-2$ mnożeń/dzieleń i tyle samo dodawania. Tak więc przy obliczaniu korzystamy ze wzoru

$$\begin{aligned}\sum_{i=-k}^{n} c_i p^i &= \Big(\ldots\Big((c_{-k}/p + c_{-k+1})/p + c_{-k+2}\Big)/p + \ldots c_{-1}\Big)/p + c_0 \\ &+ \Big(\ldots\Big((c_n * p + c_{n-1}) * p + c_{n-2}\Big) * p + \ldots + c_1\Big) * p.\end{aligned}$$

Podsumowanie

- Najczęściej używane **funkcje całkowitoliczbowe** to "podłoga" i "sufit". Mają one szereg interesujących własności i są tak ważne, że większość języków programowania implementuje te funkcje.
- W oparciu o funkcję "podłoga" można skonstrować nieskończoną reprezentację liczb rzeczywistych taką, że liczba α ma reprezentację $Spec(\alpha) = \{\lfloor\alpha\rfloor, \lfloor 2\alpha\rfloor, \lfloor 3\alpha\rfloor, \lfloor 4\alpha\rfloor, ...\}$. W wielu zastosowaniach reprezentacja ta może być interesującą alternatywą innych reprezentacji.
- Funkcje "podłoga " i "sufit" często występuje w rekurencjach dotyczących analizy funkcjonowania algorytmów czy też w dokładniejszej analizie problemu Flawiusza.
- Inna funkcja całkowitoliczbowa - operator dwuargumentowy *mod* możemy rozszerzyć do funkcji rzeczywistej. Operator ten jak i omawiane tutaj funkcje całkowitoliczbowe często występują w wielu algorytmach takich jak zamiana podstawy liczenia czy generowanie liczb pseudolosowych.

Rozdział 4

Teoria liczb

4.1. Podzielność, NWD i NWW **62**
4.2. Liczby pierwsze **65**
4.3. Reprezentacja liczb wymiernych - drzewo Sterna - Brocota **70**
4.4. Kongruencje **73**
4.5. Funkcja ϕ Eulera i algorytm RSA **76**

4.1. Podzielność, NWD i NWW

Niech $m, n \in \mathcal{Z}$. Przez $m|n$ oznaczamy fakt, że istnieje liczba całkowita k taka, że $n = mk$ (np. $-3|12$). Podobnie $m \not| n$ ($3 \not| -7$). Dodatnią całkowitą liczbę p nazywamy pierwszą, jeżeli ma dokładnie dwa różne dzielniki: 1 i p. Stąd 1 nie jest liczbą pierwszą (dzieli się tylko przez 1) ale liczbami pierwszymi są m.in. $\{2, 3, 5, 7, 11, 13, 17, 19, ...\}$. Dla liczby pierwszej p i liczby naturalnej r i całkowitej n mówimy, że p^r jest dokładnym dzielnikiem n ($p^r||n$) jeśli $p^r|n$ i $p^{r+1} \not| n$.

Twierdzenie 4.1.1. *Dla dowolnych całkowitych $a, b, c, m, n,$ oraz naturalnych d, e, p, r zachodzi:*

$$
\begin{array}{lrcl}
(i) & a|b & \Longrightarrow & (a \leq b) \wedge (\frac{b}{a}\Big|b) \wedge (\bigwedge_{k\in\mathcal{Z}} a^k|b^k) \wedge (\bigwedge_{c\in\mathcal{N}} a|bc), \\
(ii) & a|b \wedge b|c & \Longrightarrow & a|c, \\
(iii) & a|b \wedge a|c & \Longleftrightarrow & a|(bm + cn) \wedge a|c, \\
(iv) & a|b \wedge b|a & \Longrightarrow & a = \pm b, \\
(v) & a|b & \Longrightarrow & am|bm, \\
(vi) & p^r||d \wedge p^s||e & \Longrightarrow & p^{r+s}||de, \\
(vii) & p^r||d \wedge p^s||e & \Longrightarrow & p^{\min\{r,s\}}||(d + e).
\end{array}
$$

□

Definicja 4.1.1. *Największym wspólnym dzielnikiem i najmniejszą wspólną wielokrotnością liczb m i n nazywamy*

$$
\begin{aligned}
NWD(m, n) &= \max\{k : k|m \wedge k|n\}, \\
NWW(m, n) &= \min\{k : m|k \wedge n|k\}.
\end{aligned}
$$

□

Twierdzenie 4.1.2. *Dla dowolnych całkowitych $a, b, c,$ zachodzi:*

$$
\begin{array}{lrcl}
(i) & a|b \wedge a|c & \Longrightarrow & a|NWD(b, c), \\
(ii) & a|c \wedge b|c & \Longrightarrow & NWW(a, b)|c, \\
(iii) & NWD(d, e) = NWD(e \bmod d, d) & \Longleftrightarrow & d < e, \\
(iv) & NWD(a, b) = c & \Longrightarrow & NWD(\frac{a}{c}, \frac{b}{c}) = 1, \\
(v) & NWD(a + bc, b) = NWD(a, b), & & \\
(vi) & a, b, c \neq 0 & \Longrightarrow & NWD(ac, bc) = |c|NWD(a, b), \\
(vii) & NWD(a, c) = NWD(b, c) = 1 & \Longrightarrow & NWD(ab, c) = 1.
\end{array}
$$

□

Szczególnie interesująca jest własność (iii) będąca konsekwencją Twierdzenia 4.1.1. (iii) i stanowiąca podstawę rekurencyjnej definicji NWD:

$$\begin{cases} NWD(n,0) &= n, \\ NWD(n,m) &= NWD(m, n \bmod m), \end{cases}$$

która jest podstawą obliczania $NWD(n,m)$ dla dowolnych liczb całkowitych n i m, następującym algorytmem

Algorytm 4.1. Algorytm Euklidesa

```
integer function NWD(n,m: integer);
   {
    k: integer;
    if m=0 then
        NWD:=n
    else
        NWD:=NWD(m, n mod m);
    }
```

Przykład 4.1.1. *Chcemy obliczyć* $NWD(28,36)$ *wtedy mamy*

$$\begin{aligned} 28 &= 0 \cdot 36 + 28, \\ 36 &= 1 \cdot 28 + 8, \\ 28 &= 3 \cdot 8 + 4, \\ 8 &= 2 \cdot 4 + 0, \end{aligned}$$

lub w skrócie $NWD(28,36) = NWD(36,28) = NWD(28,8) = NWD(8,4) = NWD(4,0) = 4$.

Analogicznie jeśli chcemy obliczyć $NWD(-15,25)$ *to*

$$\begin{aligned} -15 &= -1 \cdot 25 + 10, \\ 25 &= 2 \cdot 10 + 5, \\ 10 &= 2 \cdot 5 + 0, \end{aligned}$$

czyli $NWD(-15,25) = NWD(25,10) = NWD(10,5) = NWD(5,0) = 5$.

□

Natomiast $NWW(n,m)$ obliczamy z tożsamości $NWW(n,m) = \frac{nm}{NWD(n,m)}$.

Uwaga 4.1.3. *Dla dowolnych liczb całkowitych* m, n *istnieją liczby całkowite* m', n' *takie, że*

$$m'm + n'n = NWD(n,m).$$

Dowód. Jeżeli $m = 0$ to bierzemy $m' = 0, n' = 1$ w przeciwnym razie stosujemy tę metodę rekurencyjnie dla $r = n \bmod m$ i m zamiast m i n. Z założenia rekurencyjnego istnieję liczby r' i m' takie, że

$$\begin{aligned}(m' - \lfloor\frac{n}{m}\rfloor r')m + r'n &= r'(n - \lfloor\frac{n}{m}\rfloor m) + m'm = r'(n \bmod m) + m'm \\ &= NWD(n \bmod m, m) = NWD(n, m).\end{aligned}$$ □

Przykład 4.1.2. *W przykładzie 4.1.1 mamy*

$$\begin{aligned}8 &= 1 \cdot 36 + (-1) \cdot 28 \\ 4 &= 28 - 3 \cdot 8 = 28 - 3 \cdot (36 - 28) = (-3) \cdot 36 + 4 \cdot 28.\end{aligned}$$ □

Następujący algorytm wylicza liczby m' i n' spełniające Uwagę 4.1.3:

Algorytm 4.2. Rozszerzony algorytm Euklidesa

```
integer function NWD(n,m,n',m': integer);
    n0, m0, r, s, new_r, new_s, reszta, iloraz : integer;
    { Dane są liczby n i m. Program zwraca NWD(n,m) i wyliczone n' oraz m'. }
    n0:=n; m0:=m;
    n':=1; m':=0;
    r:=0; s:=1;
    { Utrzymujemy n'*n+m'*m=n0 oraz r*n+s*m=m0 }
    while (m0<>0)
        {
         reszta:=n0 mod m0;
         iloraz:=⌊ n0/m0 ⌋;
         n0:=m0;
         m0:=reszta;
         new_r:=n'-iloraz*r;
         new_s:=m'-iloraz*s;
         n':=r;
         m':=s;
         r:=new_r;
         s:=new_s;
        }
    { Wtedy NWD(n,m)=n0=n'*n+m'*m }
    NWD:=n0;
```

Uwaga 4.1.4. *Zauważmy, że dla ustalonej liczby naturalnej n:*

$$\sum_{\{m \in \mathcal{N} : m|n\}} a_m = \sum_{\{m \in \mathcal{N} : m|n\}} a_{\frac{n}{m}}.$$ □

4.2. Liczby pierwsze

Jak wspomniano wcześniej liczba naturalna p jest liczbą pierwszą, jeśli ma dokładnie dwa dzielniki 1 i p. Zestawimy liczby pierwsze mniejsze od 1000 w następującej tabelce:

Tabela 4.1. Liczby pierwsze mniejsze od 1000.

2	3	5	7	11	13	17	19	23	29
31	37	41	43	47	53	59	61	67	71
73	79	83	89	97	101	103	107	109	113
127	131	137	139	149	151	157	163	167	173
179	181	191	193	197	199	211	223	227	229
233	239	241	251	257	263	269	271	277	281
283	293	307	311	313	317	331	337	347	349
353	359	367	373	379	383	389	397	401	409
419	421	431	433	439	443	449	457	461	463
467	479	487	491	499	503	509	521	523	541
547	557	563	569	571	577	587	593	599	601
607	613	617	619	631	641	643	647	653	659
661	673	677	683	691	701	709	719	727	733
739	743	751	757	761	769	773	787	797	809
811	821	823	827	829	839	853	857	859	863
877	881	883	887	907	911	919	929	937	941
947	953	967	971	977	983	991	997		

4.2.1. Podstawowe twierdzenie arytmetyki

Twierdzenie 4.2.1. Podstawowe twierdzenie arytmetyki. *Każda liczba naturalna $n \neq 1$ może być w sposób jednoznaczny przedstawiona jako iloczyn liczb pierwszych*

$$n = p_1 \cdot p_2 \cdot ... \cdot p_m = \prod_{k=1}^{m} p_k, ,$$

gdzie $p_1 \leq p_2 \leq p_3 \leq ... \leq p_m$ jest niemalejącym ciągiem liczb pierwszych. Jeżeli przyjąć, że $\prod_{k=1}^{0} p_k = 1$ to do tej własności można włączyć $n = 1$.

Dowód. Indukcja względem n.

Krok 1. Dla $n = 1$, $p_1 = 1$ twierdzenie prawdziwe.

Krok 2. Załóżmy, że twierdzenie jest prawdziwe dla wszystkich liczb naturalnych mniejszych od n. Jeśli liczba n nie ma dzielników różnych od 1 i n to jest ona liczbą pierwsz a. Jeśli ma dzielnik $k, 1 < k < n$, który jest liczbą złożoną, to z założenia indukcyjnego zarówno k jakl i n/k można przedstawić

jako iloczyny liczb pierwszych a więc i $n = k \cdot \frac{n}{k}$ można przedstawić jako iloczyn liczb pierwszych.

Załóżmy, że n nie ma jednoznacznego rozkładu (czyli ma dwa różne rozkłady. Zatem $n = p_1 p_2 ... p_m,\ p_1 \leq p_2 \leq ... \leq p_m$ oraz $n = q_1 q_2 ... q_t,\ q_1 \leq q_2 \leq ... \leq q_t$. Można przyjąć, że $p_j = q_j, j < i$, i z symetrii $p_i < q_i$. Wtedy

$$n = \underbrace{p_1 \cdot p_2 \cdot ... \cdot p_{i-1}}_{c} \overbrace{p_i \cdot ... \cdot p_m}^{a}$$

$$n = \underbrace{q_1 \cdot q_2 \cdot ... \cdot q_{i-1}}_{c} \overbrace{q_i \cdot ... \cdot q_t}^{b}$$

Mamy $a = b$ oraz $NWD(a, b) = a = b$ a wobec tego jeśli $p_i | a$ to i $p_i | b$ a w ten sposób dochodzimy do sprzeczności, bo $p_i \neq q_i$ $(p_i < q_i)$ a więc p_i nie może być żadnym z dzielników $q_i ... q_t$. □

Jeżeli zliczymy krotności występowania kolejnych liczb pierwszych (p_1 występuje d_1-razy, itd..) to powyższy wynik możemy sformułować

Twierdzenie 4.2.2. Skrócona wersja podstawowego twierdzenia arytmetyki. *Każdą liczbę naturalną n można jednoznacznie przedstawić w postaci*

$$n = \prod_{k=1}^{m} p_k^{d_k},$$

gdzie $p_1 < p_2 < ... < p_m$ jest rosnącym ciągiem liczb pierwszych a $\{d_i, 1 \leq i \leq m\}$ ciągiem dodatnich liczb naturalnych. □

Na przykład:

$$\begin{aligned} 6 &= 2 \times 3, \\ 245 &= 5 \times 7^2, \\ 10! &= 2^8 \times 3^4 \times 5^2 \times 7, \\ 68\ 718\ 821\ 377 &= (2^{17} - 1) \times (2^{19} - 1). \end{aligned}$$

Największą obecnie znaną liczbą pierwszą jest $2^{43\,112\,609} - 1$. Ma ona 12 978 189 i otrzymano ją w sierpniu 2008 roku.

Problem znalezienia przedstawienia dowolnej liczby naturalnej n w postaci iloczynu liczb pierwszych nosi nazwę problemu faktoryzacji i problem ten doczekał się wielu algorytmów. Jeżeli w faktoryzacji jedynym pierwszym składnikiem jest n to w konkluzji otrzymujemy stwierdzenie, że n jest liczbą pierwszą, w pozostałych przypadkach n nie jest liczbą pierwszą. Algorytmy

faktoryzacji służą więc też do testowania, czy dana liczba jest pierwsza, są one jednak mało wydajne. Najprostszy taki algorytm prezentujemy poniżej

Algorytm 4.3. Test dzielników

Dane: Liczba naturalna $n \geq 2$,

Wynik: j (najmniejszy pierwszy czynnik n) lub stwierdzenie, że n jest liczbą pierwszą

```
j := 2;
while j≤ √n
    {
        Jeśli j|n wtedy j jest najmniejszym pierwszym czynnikiem n
            i stop {n nie jest liczbą pierwszą}
        j := j + 1;
    }
{Nie znaleziono pierwszych czynników n więc n jest liczbą pierwszą}
```

Oczywiście koszt (ilość obrotów pętli) tego algorytmu wynosi w jagorszym przypadku $\sqrt{n}$. Można ten koszt trochę zmniejszyć rozważając dla $j \geq 3$ co drugą a nie kolejne liczby ($j = 2, 3, j + 2$). To ulepszenia ma związek z sitem Eratostenesa które omówimy w podrozdziale 4.2.3. Zwróćmy też uwagę, że nie zatrzymując algorytmu po znalezieniu najmniejszego dzielnika n, możemy łatwo otrzymać wszystkie dzielniki danej liczby n. Formalnie odpowiada to zastąpieniu instrukcji "stop" instrukcją "wydaj j i n/j".

4.2.2. Reprezentacja liczb naturalnych

Podajmy kilka kolejnych liczb pierwszych:

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, ...$$

i ponumerujmy te liczby, tak że $n(2) = 1, n(3) = 2, n(5) = 3, n(7) = 4, ...$ Z każdą liczbą naturalną n o przedstawieniu $n = \prod_{k=1}^{m} p_k^{d_k}$, kojarzymy ciąg $< 0, .., 0, d_1, 0, ..0, d_2, ... >$ gdzie odpowiednie liczby d_i znajdują się na pozycji $n(p_i),\ 1 \leq i \leq m$. Tak więc na przykład:

$$\begin{aligned} 9 &= 3^2 =< 0, 2, 0, ... >, \\ 14 &= 2 \cdot 7 =< 1, 0, 0, 1, 0, ... >, \\ 24 &= 2^3 \cdot 3 =< 3, 1, 0, ... > . \end{aligned}$$

Kolejne elementy tej reprezentacji oznaczać będziemy przez $n =< n_1, n_2, n_3, ... > .$ Łatwo widać, że wszystkie wyrazy od pewnego miejsca są równe 0, oraz że

$$\begin{aligned}(k = m \cdot n) &\iff (\forall_{p=1,2,...} k_p = m_p + n_p),\\ (k = NWD(m,n)) &\iff (\forall_{p=1,2,...} k_p = \min\{m_p, n_p\}),\\ (k = NWW(m,n)) &\iff (\forall_{p=1,2,...} k_p = \max\{m_p, n_p\}).\end{aligned}$$

Definicja 4.2.1. *Liczby $n, m \in \mathcal{Z}$ nazywamy względnie pierwszymi $m \perp n$ jeśli $NWD(n,m) = 1$.* □

Uwaga 4.2.3.

$$\begin{aligned}(m \perp n) &\iff (\forall_{p=1,2,...} \min(n_p, m_p) = 0,)\\ (m \perp n) &\iff (\forall_{p=1,2,...} n_p m_p = 0.)\end{aligned}$$ □

4.2.3. Własności i wyznaczanie liczb pierwszych

Ile jest liczb pierwszych i jak je możemy wyznaczać? Na pierwszą część pytania daje następujący wynik sformułowany w formie hipotezy przez Carla Friedricha Gaussa w 1792 roku

Twierdzenie 4.2.4. *(Vallée Poussin, Jacques Hadamard). Niech p_n będzie n-tą liczbą pierwszą a $\pi(x)$ ilością liczb pierwszych mniejszych bądź równych x. Wtedy*

$$\begin{aligned}p_n &\sim n\ln(n),\ gdy\, n \to \infty,\\ \pi(x) &\sim \frac{x}{\ln(x)}, gdy\, x \to \infty.\end{aligned}$$ □

Symbol $f(n) \sim g(n)$ oznacza, że $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$, i dokładniej będzie omawiany w podrozdziale 7.2. Jedną z metod wyznaczania kolejnych liczb pierwszych jest Algorytm 4.3 (zob. uwaga na końcu podrozdziału 4.2.1). Jest to jednak bardzo mało efektywny algorytm. Inną metodę zaproponował Eratostenes w III wieku pne. Zaproponował on następującą procedurę:

Algorytm 4.4. Sito Eratostenesa

(a) Ustaw liczby $\{2, 3, 4, 5, ..., N\}$ w rosnący ciąg i przejdź na początek tego ciągu.
(b) Weź kolejną nieskreśloną liczbę k i skreślaj jej wielokrotności $2k, 3k, \ldots,$ mniejsze od $N/2$. Postępuj tak dopóki $k \leq N/2$.
(c) Nieskreślone liczby w tym ciągu tworzą ciąg liczb pierwszych nie większych niż N

Przykład zastosowania tego algorytmu dla $N = 20$ podajemy w następującej tabelce

Tabela 4.2. Sito Eratostenesa.

2	3	~~4~~	5	~~6~~	7	~~8~~	9	~~10~~	11	~~12~~	13	~~14~~	15	~~16~~	17	~~18~~	19	~~20~~
2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~	11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~

W ten sposób otrzymujemy następujący ciąg liczb pierwszych mniejszych od 20: $\{2, 3, 5, 7, 11, 13, 17, 19\}$.

Jak widać opisana procedura jest zdecydowanie efektywniejsza od Algorytmu 4.3 jednak kosztem dużej (rzędu N) pamięci.

Na szczególną uwagę zasługuje zbiór liczb postaci $M(p) = 2^p - 1$, gdzie p jest liczbą pierwszą. Liczby te nazywamy liczbami Mersenne'a . Dla niektórych liczb pierwszych są to też liczby pierwsze. Podamy bez dowodu:

Twierdzenie 4.2.5. *Jeżeli liczba pierwsza p spełnia test Lucasa-Lehmera, to znaczy*

$$S_{p-2} \equiv 0 \ (mod\ M(p)),$$

gdzie

$$S_k = \begin{cases} 4, & gdy\ k = 0, \\ S_{k-1}^2 - 2, & gdy\ k \neq 0, \end{cases}$$

to $M(p)$ jest liczbą pierwszą. □

co prowadzi do algorytmu:

Algorytm 4.5. Testowanie liczb pierwszych Mersenne'a

Dwie funkcje boolowskie is_prime i is_mersenne_prime sprawdzają czy liczby p i $M(p)$ są pierwsze

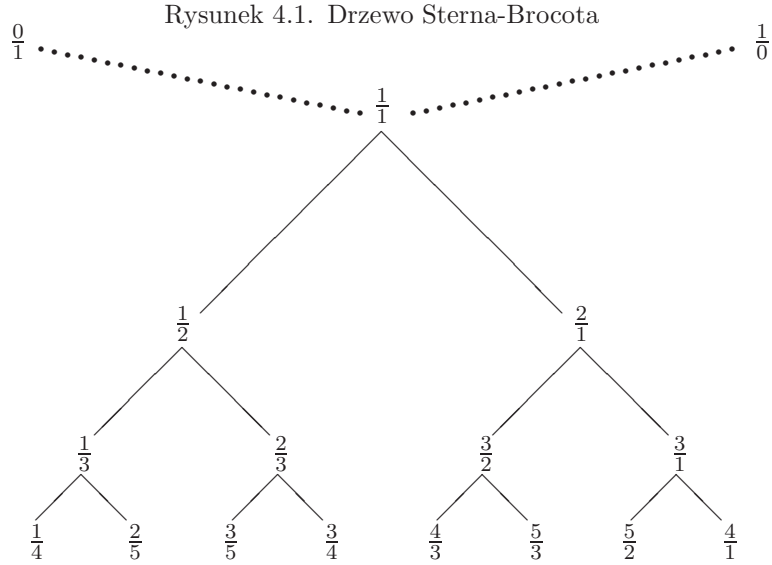
```
boolean function is_prime (p: long int):
  i : long int;
  if p=2 then
     {test Lucasa-Lehmera pracuje tylko dla nieparzystych liczb pierwszych}
     return True
  else
     { if (p<=1) or (p mod 2=0) then
           return False
       else
          { for i:=3 to sqrt(p)+1 step 2
                 { if p mod i=0 then return False; }
            return True;     }
```

```
    }
boolean function is_mersenne_prime (p: long int);
  s, m_p : long int;
  if p=2 then
     return True
  else
     { m_p:=2^p - 1;
       s:=4;
       for i:=3 to p+1
            s:=(s^2-2) mod m_p;
       return s=0;
      }
```

4.3. Reprezentacja liczb wymiernych - drzewo Sterna - Brocota

Zaczynamy od dwóch "ułamków" $\{\frac{0}{1}, \frac{1}{0}\}$. Kolejne liczby w drzewie Sterna-Brocota tworzone są według zasady, że pomiędzy $\frac{m}{n}$ a $\frac{m'}{n'}$ wstawiamy $\frac{m+m'}{n+n'}$. Każdą liczbę wymierną reprezentujemy opisując drogę jaką należy odbyć od korzenia $\frac{1}{1}$ do tej liczby określając literę L jeśli idziemy w lewo i P jeśli idziemy w prawo (zob. Rys. 4.1).

Rysunek 4.1. Drzewo Sterna-Brocota



Ciąg symboli L i P jednoznacznie określa pozycję węzła w drzewie. Na przykład $LP^2L = LPPL$ oznacza, że od $\frac{1}{1}$ idziemy w lewo do $\frac{1}{2}$, potem w prawo do $\frac{2}{3}$, następnie w prawo do $\frac{3}{4}$ i w końcu w lewo do $\frac{5}{7}$. A więc

$LP^2L = \frac{3}{4}$. Problem pojawia się z reprezentacją liczby $\frac{1}{1}$. Możemy dla niej zarezerwować inny symbol (np. I) lub traktować jako nieskończony ciąg LP^∞. Wszystkie ułamki występujące w drzewie Sterna-Brocota są nieskracalne tzn. takie, że $m \perp n$. Powyżej opisanym drzewem Sterna-Brocota można reprezentować wszystkie liczby dodatnie wymierne (wtedy reprezentacja jest skończona) oraz liczby niewymierne (wtedy reprezentacja jest nieskończona, ale może być okresowa!) z wyjątkiem liczby 1 (jak wyżej wspomniano 1 może mieć reprezentację nieskończoną). Ponadto 0 możemy reprezentować nieskończonym rowinięciem L^∞ a ∞ może być przedstawione jako P^∞.

Problem 4.3.1. *Jak zamieniać reprezentację ułamka $\frac{m}{n}$ na ciągową LP... i vice versa?* □

Niech S oznacza dowolny ciąg L i P. Ciąg S będziemy utożsamiać z macierzą $\begin{pmatrix} n & n' \\ m & m' \end{pmatrix}$ jeśli ciąg S reprezentuje ułamek $\frac{m+m'}{n+n'}$ w drzewie Sterna-Brocota (liczbę leżącą pomiędzy $\frac{m}{n}$ a $\frac{m'}{n'}$).

Zwróćmy jeszcze, że jeśli $\begin{pmatrix} n & n' \\ m & m' \end{pmatrix}$ to

$$SL = \begin{pmatrix} n & n+n' \\ m & m+m' \end{pmatrix} = \begin{pmatrix} n & n' \\ m & m' \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = S \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$SP = \begin{pmatrix} n+n' & n' \\ m+m' & m' \end{pmatrix} = \begin{pmatrix} n & n' \\ m & m' \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} = S \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

Możemy więc litery L i P utożsamiać odpowiednio z $L = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ oraz $P = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. Uwzględniając tożsamości $L^k = \begin{pmatrix} 1 & k \\ 0 & 1 \end{pmatrix}$ oraz $P^k = \begin{pmatrix} 1 & 0 \\ k & 1 \end{pmatrix}$ $k = 1, 2, 3, ...$ Pozwala to efektywnie obliczać wartość liczbową na podstawie reprezentacji Sterna-Brocota.

Przykład 4.3.1.

$$LP^2L = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 4 \\ 2 & 3 \end{pmatrix} = \frac{2+3}{3+4} = \frac{5}{7}. \quad \square$$

Również analizując tę reprezentację możemy otrzymać następujący wynik

Twierdzenie 4.3.1.

$$\frac{m}{n} = PS \iff \frac{m-n}{n} = S, \text{ jeśli } m > n,$$

$$\frac{m}{n} = LS \iff \frac{m}{n-m} = S, \text{ jeśli } m < n. \qquad \square$$

Wynik ten prowadzi do następującego algorytmu zamiany liczby wymiernej $\frac{m}{n}$ (z niekoniecznie $m \perp n$) na ciąg Sterna-Brocota S

Algorytm 4.6. Zamiana liczby wymiernej na reprezentację Sterna-Brocota

```
S:='';
while m≠n do
    if m<n then
        {S:=S & 'L'; n:=n-m}
    else
        {S:=S & 'P'; m:=m-n}
```

Przykład 4.3.2. *Zapisz $\frac{12}{5}$ w reprezentacji Sterna-Brocota.*

$$\begin{array}{lcccccc} m = & 12 & 7 & 2 & 2 & 2 & 1 \\ n = & 5 & 5 & 5 & 3 & 1 & 1 \\ S = & P & P & L & L & P & \end{array}$$

czyli $\frac{12}{5} = P^2L^2P$. $\square$

Jak wspomniano, można rozszerzyć reprezentację Sterna-Brocota na liczby niewymierne (taka reprezentacja będzie nieskończona). Na przykład

$$e = PL^0PLP^2LPL^4PLP^6LPL^8PLP^{10}LPL^{12}PL....$$

Algorytm zamiany dowolnej liczby niewymiernej α na ciąg Sterna-Brocota:

Algorytm 4.7. Zamiana liczby niewymiernej na reprezentację Sterna-Brocota

```
S:='';
while True do
    if α<1 then
        {S:=S & 'L'; α:=α/(1-α)}
    else
        {S:=S & 'P'; α:=α-1}
```

Problem 4.3.2. *Ile wynosi iloczyn liczby zapisanej w konwencji Sterna-Brocota przez liczbę w której L zostało zamienione na P i vice versa, tzn. dla $k_1, k_2, ..., k_n \in \mathcal{N}$,*

$$L^{k_1}P^{k_2}L^{k_3}...P^{k_n} \cdot P^{k_1}L^{k_2}P^{k_3}...L^{k_n} = ?$$

a ile wynosi

$$L^{k_1}P^{k_2}L^{k_3}...P^{k_n} + LP^{k_1-1}L^{k_2}P^{k_3}...L^{k_n} =?$$ □

Problem 4.3.3. *Jak będzie wyglądać drzewo Sterna-Brocota dla początkowego ciągu* $\{\frac{-1}{0}, \frac{1}{0}\}$ *i jak przeliczać reprezentację tego drzewa na reprezentację opisaną w tym podrozdziale? (wyjątkowo* $\frac{0}{0}$ *w pierwszym kroku zamieniamy na* $\frac{0}{1}$*)* □

4.4. Kongruencje

Wiadomo jaka są cechy podzielności liczb naturalnych przez $2, 3, 4, 9$, niektórzy znają cechy podzielności przez 11. Czy jednak istnieje jednolita metoda wyprowadzania wszystkich cech podzielności liczb?

Definicja 4.4.1. *Mówimy, że liczba całkowita a przystaje do liczby całkowitej b modulo m* $(a \equiv b \ (mod\ m))$ *jeżeli* $(a \ mod\ m) = (b \ mod\ m)$ *lub równoważnie* $m|(a-b)$, $(m \in \mathcal{N})$. □

Relacja przystawania modulo m jest, jak łatwo sprawdzić, zwrotna, symetryczna i przechodnia. Historycznie stosuje się do przystawania i do reszty ten sam zapis *mod*, utrzymamy tę konwencję zwracając uwagę na różne umieszczenie nawiasów w tych operacjach.

Twierdzenie 4.4.1. Własności kongruencji.

(a) $a \equiv b \ (mod\ m) \wedge c \equiv d \ (mod\ m) \Longrightarrow a \pm c \equiv b \pm d \ (mod\ m)$

(b) $a \equiv b \ (mod\ m) \wedge c \equiv d \ (mod\ m) \Longrightarrow a \cdot c \equiv b \cdot d \ (mod\ m)$

(c) $a \equiv b \ (mod\ m) \Longrightarrow a^n \equiv b^n \ (mod\ m), n \geq 1,$

(d) $ad \equiv bd \ (mod\ m) \Longrightarrow a \equiv b \ (mod\ \frac{m}{NWD(d,m)})$

(e) $a \equiv b \ (mod\ md) \Longrightarrow a \equiv b \ (mod\ m)$

(f) $a \equiv b \ (mod\ m) \wedge a \equiv b \ (mod\ n) \Longrightarrow a \equiv b \ (mod\ NWW(m,n))$

Dowód.

(a) $a-b = mk \wedge c-d = ml \Longrightarrow (a+c)-(b+d) = m(k+l) \wedge (a-c)-(b-d) = m(k-l), k, l \in \mathcal{N}$.

(b) $a-b = mk \wedge c-d = ml \Longrightarrow ca-bd = ac-bc+bc-bd = c(a-b)+b(c-d) = mck + mbl, k, l \in \mathcal{N}$.

(c) Indukcyjnie z (b).

(d) $(a-b)d = km \Longrightarrow (a-b) = k\frac{m}{NWD(d,m)}, k \in \mathcal{N}$.

(e) $a - b = mdk, k \in \mathcal{N}$.

(f) $a - b = km = ln \Longrightarrow \frac{kNWD(m,n)}{n} = \frac{lNWD(m,n)}{m} = t, k, l \in \mathcal{N} \Longrightarrow a - b = NWW(n,m)t = km = ln$. □

Twierdzenie 4.4.2. *Niech $W(x)$ będzie wielomianem o współczynnikach całkowitych:*

$$W(x) = c_n x^n + c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + ... + c_o,\ c_i \in \mathcal{Z}, i = 1, 2, ..., n.$$

Wtedy

$$a \equiv b\ (mod\ m) \Longrightarrow W(a) \equiv W(b)\ (mod\ m).$$

Dowód.
Z kongruencji

$$a \equiv b\ (mod\ m)$$

wynika (Twierdzenie 4.4.1 (c))

$$a^k \equiv b^k\ (mod\ m),\ k = 0, 1, 2, ...n$$

a uwzględniając oczywistą kongruencję

$$c_k \equiv c_k\ (mod\ m),\ k = 0, 1, 2, ...n$$

z Twierdzenia 4.4.1 (b) otrzymujemy

$$c_k a^k \equiv c_k b^k\ (mod\ m),\ k = 0, 1, 2, ...n \qquad (4.4.1)$$

i sumując (zob. Twierdzenie 4.4.1 (a)) te kongruencje otrzymujemy tezę. □

Zwróćmy uwagę, że dowolną liczbę naturalną o cyfrach $k = (c_n c_{n-1}...c_o)_{10}$ można przedstawić jako $k = W_k(10) = c_n 10^n + c_{n-1}10^{n-1} + ... + c_o$, jest to więc wielomian o współczynnikach całkowitych (zobacz podrozdział 3.5). Podobnie, dzieląc cyfry na pary można otrzymać przedstawienie: $k = V_k(100) = (c_1 c_o)_{10} + (c_3 c_2)_{10}100^1 + (c_5 c_4)_{10}100^2 + ...$ Powiedzmy, że interesuje nas cecha podzielności przez 11. Szukamy kongruencji, w której po jednej stronie występuje 10 lub jakaś potęga 10, po drugiej liczba całkowita możliwie mała co do modułu (np. $0, -1, 1, -2, 2$) a kongruencja jest wzięta modulo 11. Na przykład

$$10 \equiv -1\ (mod\ 11).$$

Z Twierdzenia 4.4.2 otrzymujemy

$$W_k(10) \equiv W_k(-1)\ (mod\ 11).$$

a z Definicji 4.4.1

$$(W_k(10)\ mod\ 11) = (W_k(-1)\ mod\ 11),$$

czyli k jest podzielne przez 11 wtedy i tylko wtedy, gdy $c_o - c_1 + c_2 - c_3 + ...$ jest podzielne przez 11. Na przykład dla liczby $k = 187926509$ mamy $W_k(-1) =$

$9-0+5-6+2-9+7-8+1=1$ nie jest podzielne przez 11 ale dla $k=72948179128$ jest $W_k(-1)=8-2+1-9+7-1+8-4+9-2+7=22$ jest podzielne przez 11 a zatem i 72948179128 jest podzielne przez 11.

Algorytm szybkiego potęgowania modulo. Załóżmy, że mamy policzyć $a^b \ mod \ n$. Naiwne postępowanie polega na wymnażaniu b-krotnie a i za każdym razem braniu reszty z dzielenia przez n. Szybszy sposób polega na analizie binarnego sposobu zapisu liczby $b=(b_{k-1},b_{k-2},...,b_o)_2$ i wykorzystaniu tożsamości:

$$a^b \, mod \, n = \prod_{i=0}^{k-1} a^{b_i 2^i} \, mod \, n.$$

Przykład 4.4.1. *Oblicz* 7^{12} mod 10.

- $12=(1100)_2$,
- *wyznaczamy wybrane potęgi* 7 *modulo* 10*:*

$$\begin{aligned} 7^2 \ mod \ 10 &= 49 \ mod \ 10 = 9, \\ 7^4 \ mod \ 10 &= 9 \cdot 9 \ mod \ 10 = 81 \ mod \ 10 = 1 \\ 7^8 \ mod \ 10 &= 1 \cdot 1 \ mod \ 10 = 1, \end{aligned}$$

- $7^{12} \ mod \ 10 = 7^8 \cdot 7^4 \ mod \ 10 = 1 \cdot 1 \ mod \ 10 = 1.$ □

Przykład 4.4.2. *Oblicz* 3^{51} mod 13.

- $51=(110011)_2$,
- *wyznaczamy wybrane potęgi* 3 *modulo* 13*:*

$$\begin{aligned} 3^2 &= 9, \\ 3^4 \ mod \ 13 &= 9 \cdot 9 \ mod \ 13 = 81 \ mod \ 13 = 3, \\ 3^8 \ mod \ 13 &= 3 \cdot 3 \ mod \ 13 = 9, \\ 3^{16} \ mod \ 13 &= 9 \cdot 9 \ mod \ 13 = 81 \ mod \ 13 = 3, \\ 3^{32} \ mod \ 13 &= 3 \cdot 3 \ mod \ 13 = 9. \end{aligned}$$

- $3^{51} \ mod \ 13 = 3^{32} \cdot 3^{16} \cdot 3^2 \cdot 3^1 \ mod \ 10 = 9 \cdot 3 \cdot 9 \cdot 3 \ mod \ 13 = 729 \ mod \ 13 = 1.$ □

Ćwiczenie 4.4.1.

(a) Korzystając z kongruencji $100 \equiv -1 \ (mod \ 101)$, *opisz cechę podzielności przez* 101.

(b) Jeśli $k=(c_n c_{n-1}...c_o)_8$ *to jakie warunki powinny spełniać cyfry w tej reprezentacji aby k było podzielne przez* 7*? (Wykorzystaj kongruencję* $8 \equiv 1 \ (mod \ 7)$*.)*

(c) Korzystając z kongruencji $1000 \equiv -1 \ (mod\ 13)$, *opisz cechę podzielności przez* 13. □

4.5. Funkcja ϕ Eulera i algorytm RSA

Definicja 4.5.1. *Dla dowolnej liczby naturalnej* n *niech* $\phi(n)$ *oznacza ilość spośród liczb* $\{1, 2, ..., n-1\}$ *tych liczb, które są względnie pierwsze z* n. □

Na przykład dla $n = 6$ spośród liczb $\{1, 2, 3, 4, 5\}$ liczby względnie pierwsze z 6 to 1 i 5 a więc $\phi(6) = 2$.

Podamy teraz sposób obliczania funkcji ϕ dla dowolnej liczby naturalnej n:

Twierdzenie 4.5.1. *Funkcja Eulera ma następujące własności:*

$$\begin{aligned} \phi(1) &= 1, \\ \phi(p^k) &= p^k - p^{k-1}, \text{ dla dowolnej liczby pierwszej } p \text{ i } k \in \mathcal{N}, \\ \phi(n \cdot m) &= \phi(n) \cdot \phi(m), \text{ jeśli } n \perp m. \end{aligned}$$

□

Ostatni warunek nazywany jest multyplikatywnością funkcji ϕ, jednak ta funkcja jest multyplikatywna **tylko** dla względnie pierwszych czynników a nie dla dowolnych. Pokażemy na przykładzie jak obliczyć funkcję Eulera w oparciu o Twierdzenie 4.5.1.

Przykład 4.5.1. *Oblicz* $\phi(720)$. *Rozkładamy* 720 *na czynniki pierwsze i otrzymujemy* $720 = 2^4 3^2 5$ *a ponieważ* $NWD(2^4, 3^2, 5) = 1$, *więc* $\phi(720) = \phi(2^4 3^2 5) = \phi(2^4)\phi(3^2)\phi(5)$ *i teraz z drugiej własności* $\phi(720) = (2^4 - 2^3)(3^2 - 3^1)(5^1 - 5^0) = (16 - 8)(9 - 3)(5 - 1) = 192$. □

Funkcja ϕ ma duże zastosowanie w kryptografii. Objaśnimy bliżej te zastosowania.

Twierdzenie 4.5.2. Twierdzenie Eulera. *Dla dowolnej liczby naturalnej* $n > 0$ *oraz* $x \in N, 0 < x \leq n - 1$ *mamy*

$$x^{\phi(n)} \equiv 1 \ (mod\ n).$$

□

oraz jego konsekwencję (dla liczb n będących pierwszymi)

Twierdzenie 4.5.3. Małe Twierdzenie Fermata (1640) *Jeśli* p *jest liczbą pierwszą a* $x \in N, x \perp p$ *to*

$$x^{p-1} \equiv 1 \ (mod\ p).$$

□

Z twierdzenia Eulera bezpośrednio otrzymujemu:

Twierdzenie 4.5.4. *Jeśli $b \equiv 1(mod\ \phi(n))$ to $a^b \equiv a(mod\ n)$.*

Dowód. Ponieważ dla pewnego $k \in N, k\phi(n) = b - 1$ więc z faktu, że $a^{\phi(n)} \equiv 1(mod\ n)$ mamy $a^{\phi(n)\frac{b-1}{\phi(n)}} \equiv 1^{\frac{b-1}{\phi(n)}}(mod\ n)$ a stąd (po wymnożeniu kongruencji stronami przez a) teza. □

Algorytm RSA (Rivest, Shamir, Adleman).

---

1. Generowanie kluczy

(i) *Znajdź dwie duże liczby pierwsze korzystając z sieci Eratostenesa dla wyszukiwania małych liczb pierwszych i testu Lukasa-Lehmera dla weryfikacji czy odpowiadające im liczby pierwsze Mersenne'a są też pierwsze (Algorytmy 4.4 i 4.5), Oznaczmy je jako p i q.*

(ii) *Oblicz $n = p \cdot q$ oraz $\phi(n) = (p-1) \cdot (q-1)$.*

(iii) *Znajdź liczbę e względnie pierwszą z $\phi(n)$ wykorzystując Algorytm Euklidesa 4.1 (w pętli wypróbowujemy kolejne liczby nieparzyste z jakiegoś podprzedziału przedziału $(1, n)$). Liczba ta powinna być nieparzysta i spełniać warunek $1 < e < n$.*

(iv) *Wykorzystując rozszerzony algorytm Euklidesa (Algorytm 4.2) znajdź liczbę d taką, że*

$$de \equiv 1\ (mod\ \phi(n)),$$

to znaczy

$$de + k\phi(n) = 1 = NWD(e, \phi(n)),$$

dla pewnej liczby całkowitej k. Jeśli w algorytmie otrzymamy liczbę d ujemną to wykorzystując fakt, że również

$$(d + l\phi(n))e + (k - le)\phi(n) = 1,$$

dla dowolnej naturalnej liczby l możemy uzyskać liczbę

(v) *Parę liczb (n, e) nazywamy kluczem publicznym a parę (n, d) kluczem prywatnym. Klucz publiczny możemy jawnie przekazywać wszystkim zainteresowanym a klucz prywatny przechowywany jest pod ścisłym nadzorem (w sejfie). Wartości p i q "niszczymy" aby nie wpadły w niepowołane ręce.*

---

2. Szyfrowanie danych kluczem publicznym (n, e) RSA

(i) *Otrzymujemy od adresata klucz publiczny w postaci pary liczb (n, e).*

(ii) *Wiadomość do zaszyfrowania zamieniamy na liczby naturalne t, które muszą spełniać nierówność $0 \leq t < n$. Można to wykonać w następujący sposób: Dzielimy komunikat (zakodowany kodami ASCII lub innymi np. UNICODE) na porcje i korzystając z algorytmu zamiany podstawy (Algorytm 3.4) zapisujemy je w układzie liczenia o podstawie n.*

(iii) *Na tak otrzymanych liczbach wykonujemy operację szyfrowania i otrzymujemy liczby*

$$c = t^e \ mod \ n,$$

wykorzystując algorytm szybkiego potęgowania modulo opisany w podrozdziale 4.4.

(iv) *Liczby c są zaszyfrowaną postacią liczb t (pisać będziemy $c = K(t)$) i przekazuje się je adresatowi wiadomoći. Klucz (n, e) umożliwia ich zaszyfrowanie, lecz nie pozwala ich rozszyfrować.*

---

3. Deszyfrowanie danych kluczem prywatnym (n, d) RSA

(i) *Otrzymaliśmy zaszyfrowaną wiadomość w postaci ciągu liczb. Posiadamy klucz publiczny (n, e) oraz konieczny (a udostępniony nam poufnie przez szyfranta) klucz prywatny (n, d). Liczby w otrzymanym ciągu muszą spełniać warunek $0 < c < n$.*

(ii) *Liczbę c przekształcamy na jej pierwotną wersję t (zapisywać będziemy $t = D(c)$) za pomocą kongruencji:*

$$t = c^d \ mod \ n,$$

wykorzystując algorytm szybkiego potęgowania modulo opisany w podrozdziale 4.4.

(iii) *Z otrzymanej liczby t odtwarzamy wg ustalonego systemu znaki tekstu. Teraz możemy odczytać przesłaną wiadomość.*

Dowód poprawności algorytmu RSA. *Ponieważ $K(t) = t^e \ mod \, n$ a $D(c) = c^d \ mod \, n$ wi ec $D(K(t)) = (t^e \ mod \, n)^d \ mod \, n = t^{de} \ mod \, n$ a ponieważ $de \equiv 1 (mod \ \phi(n))$ wi ec z Twierdzenia 4.5.4 otrzymujemy $D(K(t)) = (t^{de} mod \ n) = t$.* □

Przykład 4.5.2.

Klucz publiczny *Niech $p = 17, q = 19$, część publiczna to $n = pq = 323$. Wybieramy $e = 131$ a jak łatwo sprawdzić $NWD(e, n) = 1$.*

Klucz prywatny *Poszukujemy liczby d takiej, że $de \equiv 1 \ (mod \ \phi(n))$ tzn. $131d \equiv 1 \ (mod \ 288)$. Zastosujemy rozszerzony algorytm Euklidesa (Algorytm 4.2) otrzymując w poszczególnych obrotach pętli wartości:*

				n=131, m=288					
n0	*m0*	*n'*	*m'*	*r*	*s*	*reszta*	*iloraz*	*new_r*	*new_s*
131	*288*	*1*	*0*	*0*	*1*				
288	*131*	*0*	*1*	*1*	*0*	*131*	*0*	*1*	*0*
131	*26*	*1*	*0*	*-2*	*1*	*26*	*2*	*-2*	*1*
26	*1*	*-2*	*1*	*11*	*-5*	*1*	*5*	*11*	*-5*
1	*0*	*11*	*-5*	*-288*	*131*	*0*	*26*	*-288*	*131*

a wi ec $11*131-5*288=1$ *a stąd* $d=11$. *Zatem część prywatna klucza to* $(323, 11)$.

Kodowanie *Załóżmy, że chcemy zakodować liczbę* $t=7$ *(musi ona spełniać* $0<t<n=323$*). Szyfrujemy obliczając* $K(t)=t^e \bmod n$ *dla* $t=7$. *Wszystkie kongruencje, które rozważamy w tym i następnym punkcie są mod* 323 *(będziemy ten element opuszczać). Wykorzystujemy tutaj (i w następnym podpunkcie) algorytm szybkiego potęgowania modulo opisany na końcu podrozdziału 4.4.*

Korzystając z kongruencji $343=7^3\equiv 20$, *i* $400=20^2\equiv 77=7\cdot 11$, *oraz uwzględniając, że* $NWD(7,323)=1$, *otrzymujemy* $7^5\equiv 11$ *a stąd* $7^{131}=7\cdot(7^5)^{26}\equiv 7\cdot 11^{26}$. *Ponadto ponieważ* $1331=11^3\equiv 3\cdot 13=39$ *więc* $7^{131}\equiv 7\cdot 11^2\cdot 3^8\cdot 13^8$. *Teraz* $1521=3^2\cdot 13^2\equiv 229$ *więc* $7^{131}\equiv 7\cdot 11^2\cdot 229^4$, *a ponieważ* $52441=229^2\equiv 115$ *wieć* $7^{131}\equiv 7\cdot 11^2\cdot 115^2$. *Ponadto* $1265=11\cdot 115\equiv 296$ *wieć* $7^{131}\equiv 7\cdot 296^2=613312=1898\cdot 323+258$, *ostatecznie* $7^{121}\equiv 258\ (mod\ 323)$. *Otrzymaliśmy więc* $K(7)=258$.

Dekodowanie *Chcemy odszyfrować jakiś zaszyfrowany komunikat, powiedzmy* $K(t)=258$. *Mamy* $D(258)\equiv 258^d=258^{11}$ *a ponieważ* $66564=258^2\equiv 26$ *więc* $258^{11}\equiv 258\cdot 26^5$. *Z kolei* $676=26^2\equiv 30$ *więc* $258^{11}\equiv 258\cdot 26\cdot 30^2=258\cdot 26\cdot 900\equiv 258\cdot 26\cdot 254\equiv 7$, *stąd* $D(258)=7$. □

Ćwiczenie 4.5.1. *Zaszyfruj i odszyfruj komunikaty w schemacie RSA:*

(a) $t=2, p=3, q=5, n=?, e=5, d=?$,

(b) $t=7, p=3, q=?, n=15, e=5, d=?$,

(c) $t=23, p=17, q=19, n=323, e=?, d=13$. □

Podsumowanie.

- Teoria liczb, najstarsza dyscyplina matematyki, zajmuje się między innymi problemami podzielności liczb, wyznaczania liczb pierwszych czy faktoryzacji liczb. W informatyce spotykamy się często z algorytmami rozwiązującymi te problemy (algorytm Euklidesa, algorytmy faktoryzacji czy sito Eratostenesa).
- Dzięki podstawowemu twierdzeniu arytmetyki możemy wprowadzić interesującą reprezentację liczb naturalnych, szczególnie dogodną dla takich operacji jak mnożenie, dzielenie, NWD(,.,) czy NWW(.,.) natomiast

niezbyt dobrą dla operacji dodawania i odejmowania. Reprezentacja ta pozwala na wykonywanie działań na olbrzymich liczbach.

- Drzewo Sterna-Brocota jest interesującą, często efektywniejszą niż klasyczna formą binarnej reprezentacji liczb w pamięci komputera.
- Kongruencje pozwalają wyprowadzać szereg nowych cech podzielności liczb, a więc badać faktoryzację liczb.
- Zarówno kongruencje jak i funkcja Eulera $\phi(.)$ ma ogromne zastosowanie w kryptografii. Trudności (obliczeniowe) związane z faktoryzacją dużych liczb powodują, że znajomość dzielnika jakiejś bardzo dużej liczby może być podstawą budowy szyfru. W tym rozdziale omówiliśmy najbardziej znany sposób szyfrowania - algorytm RSA.

Rozdział 5

Kombinatoryka

5.1. Permutacje, kombinacje, wariacje **82**
5.2. Podziały zbioru i liczby Stirlinga **86**
5.3. Współczynniki dwumianowe **90**

Kombinatoryka to teoria obliczania liczby elementów zbiorów skończonych. Historycznie powstanie kombinatoryki jest związane z analizą gier hazardowych, a swój rozwój zawdzięcza rachunkowi prawdopodobieństwa, teorii grafów i teorii informacji.

5.1. Permutacje, kombinacje, wariacje

Problem 5.1.1. *Na ile sposobów można utworzyć parę chłopiec-dziewczynka ze zbioru 10-ciu chłopców i 15 dziewczynek? Rozwiązanie tego problemu wynika z prawdziwej dla każdych dwóch zbiorów X i Y tożsamości: $\overline{\overline{X \times Y}} = \overline{\overline{X}} \cdot \overline{\overline{Y}}$, zwanej* **zasadą mnożenia**. *Ilość tych sposobów to $150 = 10 \cdot 15$ gdyż, na miejscu chłopca możemy wstawić jeden z 10 elementów zbioru a na miejscu dziewczynki jeden z 15.* □

Podajmy jeszcze dwa podstawowe twierdzenia

Twierdzenie 5.1.1. *Jeżeli $\overline{\overline{X}} = k$ i $\overline{\overline{Y}} = n$, to liczba wszystkich funkcji $f : X \to Y$ jest rowna n^k.*

Dowód. Oznaczmy $X = \{1, 2, ..., k\}$. Funkcje $f : X \to Y$ są ciągami długości k o wyrazach ze zbioru Y. Każdy wyraz można wybrać na n sposobów, wszystkich więc ciągów jest n^k. □

Twierdzenie 5.1.2. *Jeżeli $\overline{\overline{X}} = k$ i $\overline{\overline{Y}} = n$, to liczba funkcji różnowartościowych $f : X \to Y$ jest równa $n^{\underline{k}} = n(n-1)...(n-k+1)$ ($n^{\underline{0}} = 1$).*

Dowód. Niech $X = \{1, 2, ..., k\}$ oraz $k \leq n$. Pierwszy wyraz ciągu można wybrać na n sposobów, drugi na $n-1$, a ogólnie i-ty wyraz można wybrać na $n-(i-1) = n-i+1$ sposobów, co dowodzi tezy. Dla $k > n$ nie ma funkcji $f : X \to Y$ różnowartościowych. □

Twierdzenie 5.1.3. *Istnieje dokładnie $\binom{n}{k}$ funkcji rosnących $f : \{1, ..., k\} \to \{1, ..., n\}$.*

Dowód. Każdy taki ciąg możemy utożsamiać ze ściśle rosnącym ciągiem $1 \leq i_1 < i_2 < i_3 < ... < i_k \leq n$, a ilość funkcji różnowartościowych z X do Y wynosi $n^{\underline{k}}$. Spośród $k!$ funkcji odwzorowujących różnowartościowo zbiór $\{1, 2, .., k\}$ na zbiór $\{i_1, i_2, i_3, ..., i_k\}$ jest dokładnie jedna funkcja ściśle rosnąca, tzn. taka, że $f(j) = i_j, 1 \leq j \leq k$. Stąd funkcji ściśle rosnących jest dokładnie $\frac{n^{\underline{k}}}{k!} = \binom{n}{k}$. □

Twierdzenie 5.1.4. *Istnieje dokładnie $\binom{n+k-1}{k}$ funkcji niemalejących takich, że $f : \{1, ..., k\} \to \{1, ..., n\}$.*

Dowód. Każdy taki ciąg $\{1 \leq i_1 \leq i_2 \leq i_3 \leq ... \leq i_k \leq n\}$ przenumerowujemy tak, że otrzymamy uporządkowany ciąg $\{1 \leq i_1 < i_2 + 1 < i_3 + 2 <$

$... < i_k + k - 1 \leq n + k - 1\}$ i teraz mamy ciąg ściśle rosnący a ilość takich ciągów, z twierdzenia 5.1.3 wynosi $\binom{n+k-1}{n-1} = \binom{n+k-1}{k}$. □

5.1.1. Permutacje

Problem 5.1.2.

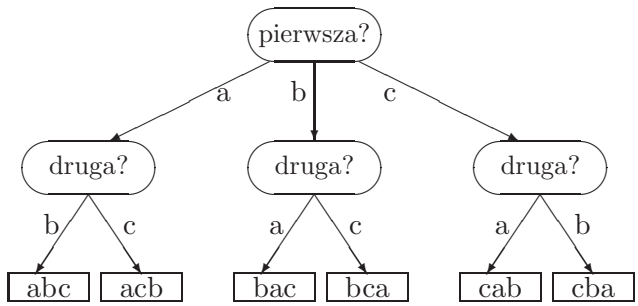
- ⋆ *Na ile sposobów można rozmieścić* 10 *osób na bardzo szerokiej kanapie?*
- ⋆ *Na ile sposobów można rozmieścić* 8 *osób wokół okrągłego stołu?*
- ⋆ *Ile różnych liczb trzycyfrowych można utworzyć z cyfr* {2,5,6}*?*
- ⋆ *Trzech panów i cztery panie wybierają się na wycieczkę gęsiego. Iloma sposobami można to zrobić, jeśli panowie nie mają sąsiadować z panami a panie z paniami?* □

Przykład 5.1.1. Permutacje bez powtórzeń zbioru $\{a, b, c\}$**.** *Mamy następujące permutacje:*

$$abc, acb, bac, bca, cab, cba.$$ □

Można je uzyskać rozważając następujące drzewo decyzyjne:

Rysunek 5.1. Permutacje zbioru $\{a, b, c\}$.



Twierdzenie 5.1.5. *Liczba permutacji zbioru* n *różnych obiektów wynosi* $P_n = n!$.

Dowód. Dowód wynika z zasady mnożenia. Ze zbioru n elementowego na pierwszym miejscu możemy wybrać jeden spośród n elementów, na drugim jeden spośród $n-1$ elementów (bo 1 element już został wybrany) itd.... Natychmiastowy dowód możemy również otrzymać z twierdzenia 5.1.2 kładąc $X = Y$ i $n = \overline{\overline{X}}$ oraz zauważając, że $n^{\underline{n}} = n!$. □

Problem 5.1.3.

- ⋆ *Dziecko bawi się czterema literkami AAMM. Ile różnych napisów może ułożyć z tych liter?*

⋆ *W urnie są trzy kule oznaczone cufrą* 1*, cztery cyfrą* 2 *i dwie cyfrą* 3*. Wyciągamy losowo wszystkie kule notując ich numery. Ile jest możliwych wyników?*

⋆ *Na ile sposobów można rozmieścić na* 10 *miejscach identyczne monety układając je orłem do góry lub reszką?* □

Przykład 5.1.2. Permutacje z powtórzeniami zbioru $\{a, a, b, b\}$**.** *Mamy następujące permutacje:*

$$aabb, abab, abba, baab, baba, bbaa.$$ □

Twierdzenie 5.1.6. *Liczba permutacji zbioru* k *obiektów o krotnościach* $n_1, n_2, ..., n_k$ *wynosi* $P_n^{n_1,n_2,...,n_k} = \frac{n!}{n_1!n_2!...n_k!}$*, gdzie* $n = n_1 + n_2 + ... + n_k$*.*

Dowód. Oznaczmy sobie następująco permutowane elementy: $a_1^{(1)}, a_2^{(1)}$, $..., a_{n_1}^{(1)}, a_1^{(2)}, a_2^{(2)}, ..., a_{n_2}^{(2)}, ..., a_1^{(k)}, a_2^{(k)}, ..., a_{n_k}^{(k)}$ gdzie elementy $a_j^{(i)}, 1 \leq j \leq n_i$ są nierozróżnialne. Ilość ustawień tak "sztucznie" rozróżnionego ciągu wynosi $n! = (n_1 + n_2 + ... + n_k)!$ a ponieważ elementy $a_j^{(i)}, 1 \leq j \leq n_i$ mogą się zmieniać na swoich pozycjach na $n_i!$ sposobów, więc ilość powtarzających się permutacji, z zasady mnożenia, wynosi $n_1!n_2!...n_k!$. □

5.1.2. Kombinacje

Problem 5.1.4.

⋆ *Na ile sposobów można wybrać* 13 *kart z talii zawierającej* 52 *karty? Kolejność wybierania kart nie jest ważna.*

⋆ 41 *szachistów rozgrywa pierwszą turę turnieju grając dokładnie raz każdy z każdym. Ile rozegrano partii ?*

⋆ *Ile różnych płaszczyzn można przeprowadzić przez* 5 *punktów w* $\Re^3$ *z których żadne* 4 *nie leżą w jednej płaszczyźnie?* □

Przykład 5.1.3. Kombinacje dwuelementowe bez powtórzeń zbioru $\{a, b, c, d\}$ **to**

$$ab, ac, ad, bc, bd, cd.$$ □

Twierdzenie 5.1.7. *Ilość kombinacji bez powtórzeń wyboru* k *elementów spośród* n *różnych elementów wynosi* $C_n^k = \binom{n}{k}$*.*

Dowód. Natychmiastowa konsekwencja Twierdzenia 5.1.3. □

Problem 5.1.5.

⋆ *Mamy cztery rodzaje owoców: jabłka, gruszki, morele i pomarańcze. Tworzymy paczki po pięć owoców w każdej. Ile można otrzymać w ten sposób różnych paczek?*

⋆ *W ciągu roku pewien klient ma wpłacić* 8 *rat miesięcznych. Iloma sposobami można to uczynić?*

⋆ *Na ile sposobów można utworzyć mieszaniny składające się z co najwyżej* 3 *farb mając do dyspozycji* 5 *farb?* □

Przykład 5.1.4. Kombinacje dwuelementowe z powtórzeniami zbioru $\{a, b, c, d\}$ *to*

$$aa, ab, ac, ad, bb, bc, bd, cc, cd, dd.$$ □

Twierdzenie 5.1.8. *Ilość kombinacji z powtórzeniami* k *elementów spośród* n *różnych elementów wynosi* $\overline{C}_n^k = \binom{n+k-1}{k}$.

Dowód. Dowód wynika z Twierdzenia 5.1.4. □

5.1.3. Wariacje

Problem 5.1.6.

⋆ *Ile można utworzyć* 3 *- kolorowych flag, gdzie kolory układane są w jednakowych poziomych pasach mając do wyboru* 6 *barw? Kolejność barw jest istotna (porównaj flagę Polski i Monako).*

⋆ *Ile można utworzyć liczb parzystych pięciocyfrowych o niepowtarzających się cyfrach?*

⋆ *Iloma sposobami można przydzielić* 4 *robotnikom* 7 *prac zakładając, że każdy z nich może wykonać tylko jedną pracę?* □

Przykład 5.1.5. Wariacje dwuelementowe bez powtórzeń zbioru $\{a, b, c, d\}$ *to*

$$ab, ba, ac, ca, ad, da, bc, cb, bd, db, cd, dc.$$ □

Twierdzenie 5.1.9. *Ilość wariacji bez powtórzeń* k *elementów spośród* n *różnych elementów wynosi* $V_n^k = \frac{n!}{(n-k)!}$.

Dowód. Z Twierdzenia 5.1.7 k elementów można wybrać ze zbioru n elementowego na $\binom{n}{k}$ sposobów a każdy wybrany ciąg można ustawić z Twierdzenia 5.1.5 na $k!$ sposobów, stąd $V_n^k = \binom{n}{k}k! = \frac{n!}{(n-k)!}$. □

Problem 5.1.7.

⋆ *Na ile sposobów cztery osoby mogą wysiąść z windy w 10 piętrowym budynku?*

⋆ *Na ile sposobów można rozmieścić* 5 *kapeluszy w* 7 *szufladach?*

⋆ *Ile można utworzyć róznych liczb* 5 *cyfrowych z cyfr* $\{0, 2, 4, 6, 8\}$*?* □

Przykład 5.1.6. Wariacje dwuelementowe z powtórzeniami *zbioru* $\{a, b, c, d\}$ to

$$aa, ab, ba, ac, cc, ca, ad, dd, da, bc, cb, bd, bb, db, cd, dc.$$ □

Twierdzenie 5.1.10. *Ilość wariacji z powtórzeniami k elementów spośród n różnych elementów wynosi* $\overline{V}_n^k = n^k$.

Dowód. Dowód jest konsekwencją Twierdzenia 5.1.1. □

5.2. Podziały zbioru i liczby Stirlinga

5.2.1. Liczby Stirlinga

Problem 5.2.1. *Na ile sposobów można podzielić zbiór* $\{a, b, c, d, e\}$ *tak, aby zawsze były dwa zbiory liczebności (mocy) 1 i jeden zbiór mocy 3. Taki podział nazywamy podziałem* $1^2 3^1$.

Rozwiązanie. *Jest 10 takich podziałów:*

$$\begin{array}{lll} \{\{a\}\{b\}\{c,d,e\}\} & \{\{a\}\{c\}\{b,d,e\}\} & \{\{a\}\{d\}\{b,c,e\}\} \\ \{\{a\}\{e\}\{b,c,d\}\} & \{\{b\}\{c\}\{a,d,e\}\} & \{\{b\}\{d\}\{a,c,e\}\} \\ \{\{b\}\{e\}\{a,c,d\}\} & \{\{c\}\{d\}\{a,b,e\}\} & \{\{c\}\{e\}\{a,b,d\}\} \\ \{\{d\}\{e\}\{a,b,c\}\} & & \end{array} \tag{5.2.1}$$

□

Twierdzenie 5.2.1. *Liczba podziałów zbioru n elementowego takich, że*
jest λ_1 *klas o liczebności 1*
jest λ_2 *klas o liczebności 2*
$\vdots$
jest λ_n *klas o liczebności n*
gdzie $n = 1 \cdot \lambda_1 + 2 \cdot \lambda_2 + ... + n \cdot \lambda_n$ *wynosi*

$$P^{\lambda_1,\lambda_2,...,\lambda_n} = \frac{n!}{\lambda_1!\lambda_2!...\lambda_n!(1!)^{\lambda_1}(2!)^{\lambda_2}...(n!)^{\lambda_n}}.$$

Dowód. Potraktujmy permutację typu $(\lambda_1, \ldots, \lambda_n)$, jako permutację zbioru $\{1, 2, ..., n\}$ według następującego wzorca:

$$\underbrace{(\bullet)\ldots(\bullet)}_{\lambda_1\ razy}\underbrace{(\bullet\bullet)\ldots(\bullet\bullet)}_{\lambda_2\ razy}\ldots\ldots\underbrace{(\bullet\ldots\bullet)}_{\lambda_n\ razy\ (\alpha_n\leq 1)}.$$

W miejsce k kropek możemy wstawić k-elementów na $k!$ sposobów. Jednak w ten sposób otrzymamy wielokrotnie te same permutacje. Każdy cykl i-elementowy możemy zadać na i sposobów (rozpoczynając od różnych elementów). Dodatkowo, zwróćmy uwagę, że w naszym wzorcu dopuszczamy różną kolejność cykli o tej samej długości. λ_i takich samych cykli i-elementowych może być wybranych na $\lambda_i!$ sposobów. □

Definicja 5.2.1. *Przez $S(n,k)$ oznaczamy ilość podziałów zbioru n-elementowego na k-klas i nazywamy ją liczbą Stirlinga II-go rodzaju.*

Twierdzenie 5.2.2. *Liczby Stirlinga drugiego rodzju spełniają następujące równanie rekurencyjne:*

$$\begin{cases} S(n+1,k) &= S(n,k-1)+kS(n,k), \quad dla\ 1 \le k < n, \\ S(n,1) &= S(n,n)=1. \end{cases} \tag{5.2.2}$$

□

Ćwiczenie 5.2.1. *Udowodnij tożsamość:*

$$x^n = \sum_{k=0}^{n} S(n,k)x^{\underline{k}}.$$

□

Problem 5.2.2. *Niech $[ABC]$ oznacza cykl elementów ABC. Oczywiście $[ABC]=[BCA]=[CAB]$ jednak $[ABC] \neq [ACB]$. Na ile sposobów można podzielić zbiór $\{a,b,c,d\}$ na dwa cykle?*

Rozwiązanie. *Jest 11 takich podziałów:*

[a,b,c][d]	[a,b,d][c]	[a,c,d][b]
[b,c,d][a]	[a,c,b][d]	[a,d,b][c]
[a,d,c][b]	[b,d,c][a]	[a,b][c,d]
[a,c][b,d]	[a,d][b,c]	

□

Definicja 5.2.2. *Przez $s(n,k)$ oznaczamy ilość podziału zbioru n-elementowego na k cykli i nazywamy ją liczbą Stirlinga I-go rodzaju.* □

Twierdzenie 5.2.3. *Liczby Stirlinga pierwszego rodzaju spełniają następujące równanie rekurencyjne:*

$$\begin{cases} s(n+1,k) &= s(n,k-1)+ns(n,k), \quad dla\ 1 \le k < n \\ s(n,1) &= (n-1)!, \\ s(n,n) &= 1. \end{cases} \tag{5.2.3}$$

□

Ćwiczenie 5.2.2. *Udowodnij tożsamość:*

$$x^{\underline{n}} = \sum_{k=0}^{n} s(n,k)(-1)^{n-k}x^k.$$

□

5.2.2. Podziały liczby

Podział liczby n na k składników to przedstawienie n w postaci sumy $a_0 + \ldots + a_{k-1} = n$, gdzie $1 \leq a_0 \leq a_1 \leq \ldots \leq a_{k-1}, a_i \in \mathcal{N}, 0 \leq i < k$. Liczbę podziałów n na k składników oznaczamy przez $P(n,k)$.

Przykład 5.2.1. *Podziały* 8 *na* 4 *składniki są następujące:*

$$1{+}1{+}1{+}5, \qquad 1{+}1{+}2{+}4, \qquad 1{+}1{+}3{+}3 \qquad 1{+}2{+}2{+}3 \qquad 2{+}2{+}2{+}2,$$

a więc $P(8,4) = 5$. □

Nie potrafimy podać nawet rekurencyjnego wzoru na $P(n,k)$, ale podamy tutaj pewne własności tej funkcji.

Uwaga 5.2.4. *Dla* $n, k \in \mathcal{N} - \{0\}$

$$P(n,1) = 1, \quad P(n,2) = \left\lfloor \frac{n}{2} \right\rfloor, \quad P(n,n) = 1, \quad P(n,k) = 0, \ \textit{dla}\ n < k,$$
$$\frac{1}{k!}\binom{n-1}{k-1} \leq P(n,k) \leq \binom{n-1}{k-1}.$$ □

Uwaga 5.2.5. *Dla* $n, k \in \mathcal{N}$

$$P(n,k) \leq \frac{1}{k!}\binom{n + \binom{k}{2} - 1}{k-1}.$$

Dowód. Dla podziału $n = a_0 + \ldots + a_{k-1}$ definiujemy $b_i = a_i + (k - 1 - i)$, dla $0 \leq i \leq k-1$. Zauważmy, że wszystkie liczby b_i są różne oraz $b_0 + \ldots + b_{k-1} = n + \frac{k(k-1)}{2}$. A zatem podziałów liczby n na k jest tyle samo co podziałów liczby $n + \binom{k}{2}$ na k parami różnych składników. Każdy podział $n + \binom{k}{2}$ na k parami różnych składników generuje dokładnie $k!$ rozwiązań równania

$$x_0 + \ldots + x_{k-1} = n + \binom{k}{2},$$

gdzie $x_i > 0$. Wiemy zaś, że to ostatnie równanie posiada co najwyżej $\binom{n+\binom{k}{2}-1}{k-1}$ rozwiązań. A zatem ciągów $\{b_i\}$, a tym samym podziałów n na k składników, jest co najwyżej $\frac{1}{k!}\binom{n+\binom{k}{2}-1}{k-1}$. □

Uwaga 5.2.6. *Dla dowolnego* $k \in \mathcal{N}$

$$\lim_{n\to\infty} \frac{P(n,k)}{n^{k-1}} = \frac{1}{k!(k-1)!}.$$ □

5.2.3. Klasyfikacja podziałów

Rozważając różne podziały obiektów na różne kategorie widać, że czasem kolejność kategorii odgrywa rolę, a czasem nie, czasem kolejność obiektów danej kategorii odgrywa rolą, a czasem nie. Często poza całkowitą liczbą konfiguracji istotna jest także liczba konfiguracji z wyłącznie niepustymi kategoriami. Poniżej zestawiamy ilości wszystkich możliwych typów klasyfikacji pewnego zbioru obiektów na różne typy kategorii:

Twierdzenie 5.2.7.

(i) Klasyfikacji rozróżnialnych obiektów na rozróżnialne kategorie jest $\overline{V}_n^k$ *czyli* k^n*. Takich klasyfikacji na dokładnie* k *kategorii jest* $k!S(n,k)$*.*

(ii) Klasyfikacji rozróżnialnych obiektów na nierozróżnialne kategorie jest $\sum_{i=1}^{k} S(n,i)$*. Oczywiście gdy wszystkie kategorie są niepuste, to zbiór obiektów jest podzielony na dokładnie* k *bloków. Liczba takich konfiguracji to* $S(n,k)$*.*

(iii) Klasyfikacji nierozróżnialnych obiektów na rozróżnialne kategorie jest tyle ile podziałów liczby n *na sumę* $n = x_0 + \ldots + x_{k-1}$ *liczb naturalnych* x_i*. Liczba rozwiązań takiego równania to* $\binom{n+k-1}{k-1}$*. A jeśli kategorii, czyli składników w rozkładzie* $n = x_0 + \ldots + x_{k-1}$*, ma być dokładnie* k*, to zliczamy jedynie rozwiązania spełniające dodatkowo* $x_0, \ldots, x_{k-1} \geq 1$*, a tych jest* $\binom{n-1}{k-1}$*.*

(iv) Klasyfikacji nierozróżnialnych obiektów na nierozróżnialne kategorie jest $P(n,k)$ *a jeśli dopuszczamy puste kategorie to liczba konfiguracji wynosi* $\sum_{i=1}^{k} P(n,i)$*.* □

Przykład 5.2.2. *Rozważmy klasyfikacje* $n = 3$ *obiektów, które jak rozważać będziemy jako rozróżnialne to oznaczać będziemy przez* $\{a,b,c\}$ *a jak nierozróżnialne to* $\{a,a,a\}$ *na* $k = 2$ *klasy (jak rozróżnialne to oznaczać będziemy przez* $\{1,2\}$ *lub* K_1, K_2*).*

(i) Z równania rekurencyjnego (5.2.2) mamy $S(3,2) = S(2,1) + 2S(2,2) = 1 + 2 = 3$*, a stąd klasyfikacji rozróżnialnych obiektów na rozróżnialne kategorie jest* $2^3 = 8$ *a na dokładnie* 2 *kategorie jest* $2!S(3,2) = 6$*. Są to klasyfikacje:*

Na 2 kategorie		*Na dokładnie 2 kategorie*	
$\{a,b,c\}$	$\{a,b,c,\}$	$\{a,b,c\}$	$\{a,b,c,\}$
1, 1, 1	1, 1, 2	1, 1, 2	1, 2, 1
1, 2, 1	1, 2, 2	1, 2, 2	2, 1, 1
2, 1, 1	2, 1, 2	2, 1, 2	2, 2, 1
2, 2, 1	2, 2, 2		

(ii) Jest $S(3,1)+S(3,2) = 1+3 = 4$ *klasyfikacji rozróżnialnych* 3 *obiektów na nierozróżnialne* 2 *kategorie i* $S(3,2) = 3$ *na dokładnie* 2 *kategorie. Są to:*

Na 2 kategorie		*Na dokładnie 2 kategorie*	
$\{a,b,c\}$	$\{a,b\},\{c,\}$	$\{a,b\},\{c\}$	$\{a,c\},\{b,\}$
$\{a\},\{b,c\}$	$\{a,c\},\{b\}$	$\{a\},\{b,c\}$	

(iii) Jest $\binom{3+2-1}{2-1} = 4$ *klasyfikacji nierozróznialnych obiektów na rozróżnialne kategorie oraz* $\binom{3-1}{2-1} = 2$ *na dokładnie* 2 *kategorie, są to*

Na 2 kategorie		*Na dokładnie 2 kategorie*
$\{a,a,a\} \in K_1$	$\{a,a\} \in K_1, \{a\} \in K_2$	$\{a\} \in K_1, \{a,a\} \in K_2$
$\{a\} \in K_1, \{a,a\} \in K_2$	$\{a,a,a\} \in K_2$	$\{a,a\} \in K_1, \{a,\} \in K_2$

(iv) Ponieważ $P(3,2) = 1$ *(rozkład* $3 = 1+2$*) oraz* $P(3,1) = 1$ *(rozkład* $3 = 3$*) więc są dwa rozkłady nierozróżnialnych* 2 *obiektów na nierozróżnielne kategorie:* $\{a,a,a\}$ *oraz* $\{a,a\},\{a\}$ *przy czym jeden (ten drugi) na dokładnie* 2 *kategorie.* □

5.3. Współczynniki dwumianowe

Jak można rozszerzyć definicję symbolu Newtona $\binom{n}{k} = \frac{n!}{k!(n-k)!} = \frac{n^{\underline{k}}}{k!}$ na przypadek wszystkich całkowitych k i rzeczywistych n? Wykorzystujemy ostatnią równość i fakt, że $r^{\underline{k}}$ można zdefiniować dla liczb rzeczywistych r i kładziemy dla $r \in \Re, k \in \mathcal{Z}$:

$$\binom{r}{k} = \begin{cases} \frac{r(r-1)(r-2)\ldots(r-k+1)}{k(k-1)(k-2)\ldots 2\cdot 1}, & k \in \mathcal{Z}_+ \cup \{0\}, \\ 0, & k \in \mathcal{Z}_-, \end{cases} = \begin{cases} \frac{r^{\underline{k}}}{k!}, & k \in \mathcal{Z}_+ \cup \{0\}, \\ 0, & k \in \mathcal{Z}_-, \end{cases}$$

($r^{\underline{0}} = 1$.)

Dla "zwykłego" symbolu Newtona $\binom{n}{k}$ zachodzi tożsamość

$$\binom{n}{k-1} + \binom{n}{k} = \binom{n+1}{k+1}.$$

Czy ta równość jest prawdziwa również dla uogólnionego symbolu Newtona? Okazuje się, że tak (prosty dowód wynika bezpośrednio z definicji):

$$\binom{r}{k-1} + \binom{r}{k} = \binom{r+1}{k+1}.$$

Dzięki temu możemy skonstruować trójkąt Pascala zawierający kolejne wartości symbolu Newtona.

Tabela 5.1. Trójkąt Pascala.

r	$\binom{r}{0}$	$\binom{r}{1}$	$\binom{r}{2}$	$\binom{r}{3}$	$\binom{r}{4}$	$\binom{r}{5}$	$\binom{r}{6}$	$\binom{r}{7}$	$\binom{r}{8}$	$\binom{r}{9}$	$\binom{r}{10}$
-10	1	-10	55	-220	715	-2002	5005	-11440	24310	-48620	92378
-9	1	-9	45	-165	495	-1287	3003	-6435	12870	-24310	43758
-8	1	-8	36	-120	330	-792	1716	-3432	6435	-11440	19448
-7	1	-7	28	-84	210	-462	924	-1716	3003	-5005	8008
-6	1	-6	21	-56	126	-252	462	-792	1287	-2002	3003
-5	1	-5	15	-35	70	-126	210	-330	495	-715	1001
-4	1	-4	10	-20	35	-56	84	-120	165	-220	286
-3	1	-3	6	-10	15	-21	28	-36	45	-55	66
-2	1	-2	3	-4	5	-6	7	-8	9	-10	11
-1	1	-1	1	-1	1	-1	1	-1	1	-1	1
0	1	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0
2	1	2	1	0	0	0	0	0	0	0	0
3	1	3	3	1	0	0	0	0	0	0	0
4	1	4	6	4	1	0	0	0	0	0	0
5	1	5	10	10	5	1	0	0	0	0	0
6	1	6	15	20	15	6	1	0	0	0	0
7	1	7	21	35	35	21	7	1	0	0	0
8	1	8	28	56	70	56	28	8	1	0	0
9	1	9	36	84	126	126	84	36	9	1	0
10	1	10	45	120	210	252	210	120	45	10	1

Twierdzenie 5.3.1. Własności uogólnionego symbolu Newtona. *Dla $k \in \mathcal{Z}$:*

$$\begin{aligned}
\binom{n}{k} &= \binom{n}{n-k}, && n \in \mathcal{N} \text{ tylko!}, && \textit{symetria,}\\
k\binom{r}{k} &= r\binom{r-1}{k-1}, && r \in \mathcal{R}, && \textit{pochłanianie,}\\
(r-k)\binom{r}{k} &= r\binom{r-1}{k-1}, && r \in \mathcal{R}, && \textit{inne pochłanianie,}\\
\binom{r}{k} &= \binom{r-1}{k} + \binom{r-1}{k-1}, && r \in \mathcal{R}, && \textit{reguła dodawania,}\\
\binom{r}{k} &= (-1)^k\binom{k-r-1}{k}, && r \in \mathcal{R}, && \textit{negowanie górnego indeksu.}
\end{aligned}$$

Dowód. Ostatnia własność. Dla $k \geq 0$ mamy $r^{\underline{k}} = r(r-1)(r-2)...(r-k+1) = (-1)^k(-r)(1-r)(2-r)...(k-r-1) = (-1)^k(k-r-1)^{\underline{k}}$, dla $k < 0$ obie strony są równe zero. □

Współczynniki dwumianowe spełniają szereg bardzo interesujących zależności. Na przykład, jeżeli w Twierdzeniu 2.7.4 podzielimy obie strony przez $m!$, to otrzymamy dla $m \geq 0$

$$\sum_{k=0}^{n-1}\binom{k}{m} = \binom{n}{m+1}.$$

Z drugiej strony, z powyższej sumy, możemy uzyskać następujący wynik:

$$\sum_{k\leq n}\binom{m+k}{k} = \sum_{-m\leq k\leq n}\binom{m+k}{k} = \sum_{-m\leq k\leq n}\binom{m+k}{m}$$

$$= \sum_{0\le k\le m+n} \binom{k}{m} = \binom{m+n+1}{m+1} = \binom{m+n+1}{n}.$$

Bezpośrednio z definicji można stwierdzić, że

$$\binom{r}{m}\binom{m}{k} = \binom{r}{k}\binom{r-k}{m-k},\ k,m \in \mathcal{Z}, r \in \mathcal{R}.$$

Innym źródłem pomagającym analizować sumy i rekurencje zawierające symbol Newtona jest:

Twierdzenie 5.3.2. *Dla $k, m \in \mathcal{Z}, r, x, y \in \mathcal{R}$, zachodzi*

$$\begin{aligned}\sum_{k\le m} \binom{r}{k}(-1)^k &= (-1)^m \binom{r-1}{m},\\ \sum_{k\le m} \binom{m+r}{k} x^k y^{m-k} &= \sum_{k\le m} \binom{-r}{k}(-x)^k (x+y)^{m-k}.\end{aligned}$$

Dowód. Pierwsza własność jest efektem negowania górnego indeksu a druga może być dowiedziona indukcyjnie względem m. □

Zwłaszcza ta druga własność może być źródłem ciekawych wzorów, np. dla $x = -1, y = 1$ otrzymujemy:

$$\sum_{k\le m} \binom{m+r}{k}(-1)^k = \binom{-r}{m},$$

a dla $x = y = 1, r = m+1$,

$$\begin{aligned}\sum_{k\le m} \binom{2m+1}{k} &= \sum_{k\le m} \binom{-m-1}{k}(-1)^k 2^{m-k}\\ &= \sum_{k\le m} \binom{m+k}{k} 2^{m-k}\end{aligned}$$

a ponieważ z symetrii trójkąta Pascala

$$\sum_{k\le m} \binom{2m+1}{k} = \frac{1}{2} \sum_{0\le k\le 2m+1} \binom{2m+1}{k} 1^k 1^{2m-k} = \frac{1}{2} 2^{2m+1} = 2^{2m}$$

a stąd

$$\sum_{k\le m} \binom{m+k}{k} 2^{m-k} = 2^m.$$

Wykorzystaliśmy w powyższym rozważaniu wzór na dwumian Newtona: $(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^k b^{n-k}$. Czy ten wzór ma swój odpowiednik dla uogólnionego symbolu Newtona?

Twierdzenie 5.3.3. Wzór na dwumian Newtona. *Niech x, y będą dwoma dowolnymi liczbami rzeczywistymi. Jeśli $r \leq 0$ jest liczbą całkowitą lub $|\frac{x}{y}| < 1$ to*

$$(x+y)^r = \sum_{k \in \mathcal{Z}} \binom{r}{k} x^k y^{r-k}. \qquad \square$$

Ćwiczenie 5.3.1. *Oblicz*

$$\sqrt[3]{9} \sum_{k \in \mathcal{Z}} \binom{1/3}{k} (-1)^k 3^{-2k}.$$

Rozwiązanie *Zauważmy, że*

$$\sqrt[3]{9} \sum_{k \in \mathcal{Z}} \binom{1/3}{k} (-1)^k 3^{-2k} = \sum_{k \in \mathcal{Z}} \binom{1/3}{k} (-1)^k 9^{1/3-k} = (-1+9)^{1/3} = \sqrt[3]{8} = 2.$$

$\square$

Omówmy teraz trzy ciekawe chwyty stosowane przy obliczaniu sum zawierających symbol Newtona:

Reguła podwajania. Poprzez uporządkowanie czynników możemy otrzymać następującą tożsamość

$$r^{\underline{k}} (r - \frac{1}{2})^{\underline{k}} = \frac{(2r)^{\underline{2k}}}{2^{2k}}, \ k \in \mathcal{N}.$$

Dzieląc stronami przez $k!^2$ otrzymujemy

$$\binom{r}{k}\binom{r-1/2}{k} = \binom{2r}{2k}\binom{2k}{k}/2^{2k}, \ k \in \mathcal{Z}. \tag{5.3.4}$$

Regułę podwajania stosuje się w przypadku, gdy mamy r szczególnej postaci $r = n - 1/2, n \in \mathcal{Z}$, wtedy możemy ze wzoru (5.3.4) przejść do innych symboli Newtona.

Metody różnicowe. Wiemy z Uwagi 2.7.1 że

$$\triangle^n f(x) = \sum_{k=0}^{n} \binom{n}{k} (-1)^{n-k} f(x+k), \ n \in \mathcal{N}, x \in \Re. \tag{5.3.5}$$

Na przykład łatwo wykazać z Twierdzenia 2.7.2, że

$$\triangle^n ((x-1)^{\underline{-1}}) = (-1)^n n! (x-1)^{\underline{-n}},$$

zatem dla $x \notin \{0, -1, -2, ..., -n\}$ otrzymujemy

$$\sum_{k=0}^{n} \binom{n}{k} \frac{(-1)^k}{x+k} = n!(x-1)^{\underline{-n}} = x^{-1} \binom{x+n}{n}^{-1}. \tag{5.3.6}$$

Reguła odwracania.

$$g(n) = \sum_{k=0}^{n} \binom{n}{k} (-1)^k f(k) \Longleftrightarrow f(n) = \sum_{k=0}^{n} \binom{n}{k} (-1)^k g(k). \tag{5.3.7}$$

dla dowolnych funkcji f, g.

Dowód (5.3.7). Ponieważ z Twierdzenia 5.3.3 dla $x = -1, y = 1, r = n$ otrzymujemy

$$\sum_{k} \binom{m}{k} (-1)^k = \begin{cases} 1, & n = 0, \\ 0, & n \in \mathcal{N} \backslash \{0\}, \end{cases}$$

więc

$$\begin{aligned} \sum_{k} \binom{n}{k} (-1)^k g(k) &= \sum_{k} \binom{n}{k} (-1)^k \sum_{j} \binom{k}{j} (-1)^j f(j) \\ &= \sum_{j} f(j) \sum_{k} \binom{n}{k} (-1)^{k+j} \binom{k}{j} \\ &= \sum_{j} f(j) \sum_{k} \binom{n}{j} (-1)^{k+j} \binom{n-j}{k-j} \\ &= \sum_{j} f(j) \binom{n}{j} \sum_{k} (-1)^k \binom{n-j}{k} \\ &= \sum_{j} f(j) \binom{n}{j} [n-j=0] = f(n). \end{aligned}$$ □

Przykład 5.3.1. *Rozwiązać rekurencję dla pewnej ustalonej liczby naturalnej m*

$$n + \frac{(-1)^n}{\binom{m}{n}} + 3^n + \frac{1}{n+3} = \sum_{k=0}^{n} \binom{n}{k} (-1)^k F_k, \ n \geq 0.$$

Rozwiązanie. *Z reguły pochłaniania*

$$\begin{aligned} \sum_{k=0}^{n} \binom{n}{k} (-1)^k k &= n \sum_{k=0}^{n} \binom{n-1}{k-1} (-1)^k = -n \sum_{k=1}^{n-1} \binom{n-1}{k} (-1)^k \\ &= -n(0-1) = n, \end{aligned} \tag{5.3.8}$$

oraz

$$\sum_{k=0}^{n}\binom{n}{k}(-1)^k\frac{(-1)^k}{\binom{m}{k}} = \sum_{k=0}^{n}\frac{\binom{n}{k}}{\binom{m}{k}} = \sum_{k=0}^{n}\frac{\binom{m-k}{n-k}}{\binom{m}{n}} = \binom{m}{n}^{-1}\sum_{k=0}^{n}\binom{m-n+k}{k}$$
$$= \frac{\binom{m+1}{n}}{\binom{m}{n}} = \frac{m+1}{m+1-n}. \tag{5.3.9}$$

Z Twierdzenia 5.3.3 mamy

$$\sum_{k=0}^{n}\binom{n}{k}(-1)^k 3^k = (3-1)^n = 2^n, \tag{5.3.10}$$

a z (5.3.6) otrzymujemy

$$\sum_{k=0}^{n}\binom{n}{k}(-1)^k\frac{1}{k+3} = 3^{-1}\binom{n+3}{3}^{-1}. \tag{5.3.11}$$

Uwzględniając (5.3.8)-(5.3.11) otrzymujemy

$$F_n = n + \frac{m+1}{m+1-n} + 2^n + \frac{1}{3\binom{n+3}{3}},\ n \geq 0. \qquad \square$$

Ćwiczenie 5.3.2. *Udowodnij:*

$$\sum_{k\in\mathcal{Z}}\binom{r}{m+k}\binom{s}{n-k} = \binom{r+s}{m+n},\ m,n\in\mathcal{Z}, r,s\in\mathcal{R},$$
$$\sum_{k\in\mathcal{Z}}\binom{t}{m+k}\binom{s}{n+k} = \binom{t+s}{t-m+n},\ m,n\in\mathcal{Z}, s\in\mathcal{R}, t\in\mathcal{Z}_+\cup\{0\},$$
$$\sum_{k\in\mathcal{Z}}\binom{t}{m+k}\binom{s+k}{n}(-1)^k = (-1)^{t+n}\binom{s-m}{n-t},\ m,n,t\in\mathcal{Z}, t>0,$$
$$\sum_{k\leq l}\binom{l-k}{m}\binom{s}{k-n}(-1)^k = (-1)^{l+m}\binom{s-m-1}{l-m-n},\ m,n,l\in\mathcal{Z}, l,m,n\leq 0,$$
$$\sum_{0\leq k\leq l}\binom{l-k}{m}\binom{q+k}{n} = \binom{l+q+1}{m+n+1},\ l,m\geq 0, n\leq q\leq 0, l,m,n,q\in\mathcal{Z},$$
$$\sum_{k\in\mathcal{Z}}\binom{r}{k}\binom{s}{n-k} = \binom{r+s}{n},\ n\in\mathcal{Z}, r,s\in\mathcal{R}. \qquad \square$$

Podsumowanie.

- Kombinatoryka zajmuje się problemami podziału zbiorów a zwłaszcza wyznaczania ilości tych podziałów. Duże znaczenie kombinatoryki wynika z faktu, że jest ona podstawą rachunku prawdopodobieństwa. W algorytmach często mamy do czynienia z losowym ustawieniem danych i analiza kombinacji możliwych ustawień jest niezbędna do prawidłowej analizy działania algorytmu.
- Różne ustawienia ciągów nazywamy permutacjami, wybór elementów ze zbioru, bez uwzględniania kolejności kombinacjami, a z uwzględnianiem kolejności wariacjami. Podziały to analiza rozkładu liczby na sumę liczb naturalnych lub zbioru na klasy o określonych liczebnościach.
- Kluczowe pojęcie w kombinatoryce odgrywa symbol Newtona. Można go uogólnić na pary liczb rzeczywistych i całkowitych. Symbol Newtona często występuje w różnych rekurencjach, sumach i innych zależnościach.

Rozdział 6

Funkcje tworzące

6.1. Splot **98**
6.2. Rozwiązywanie rekurencji za pomocą funkcji tworzących **100**
6.3. Algebra wielomianów rzeczywistych **106**

Rozważmy szereg (zob. dwumian Newtona):

$$(1+z)^r = \sum_{k\geq 0} \binom{r}{k} z^k = \binom{r}{0} z^0 + \binom{r}{1} z^1 + \binom{r}{2} z^2 + ... + \binom{r}{\lfloor r \rfloor} z^{\lfloor r \rfloor} + ..., r \in \Re. \tag{6.0.1}$$

Z Twierdzenia 5.3.3 szereg ten jest zbieżny w kole jednostkowym $|r| < 1$. Ogólnie szeregi potęgowe są zbieżne w 0 i pewnym otoczeniu zera. W tym rozdziale nie będziemy rozważać obszaru zbieżności szeregów, trzeba zdawać sobie sprawę, że wymieniane tutaj szeregi są sumowalne tylko dla pewnego przedziału argumentów.

Funkcję $A(z) = (1+z)^r$ nazywamy funkcją tworzącą ciągu $[\binom{r}{0}, \binom{r}{1}, \binom{r}{2}, \ldots]$.

Definicja 6.0.1. *Dla nieskończonego ciągu* $[a_o, a_1, a_2, ...]$ *funkcją tworzącą tego ciągu nazywamy funkcję*

$$A(z) = a_o + a_1 z + a_2 z^2 + ... = \sum_{k\geq 0} a_k z^k.$$

Jeżeli $A(z)$ *jest funkcją tworzącą to przez* $[z^k]A(z)$ *oznaczać będziemy współczynnik przy* z^k *w rozwinięciu funkcji* $A(z)$ *w szereg potęgowy.* □

6.1. Splot

Definicja 6.1.1. *Dla dwóch ciągów* $[a_o, a_1, a_2, ...]$ *oraz* $[b_o, b_1, b_2, ...]$ *odpowiadających funkcjom tworzącym* $A(z)$ *i* $B(z)$ *splot jest to ciąg odpowiadający funkcji tworzącej* $C(z) = A(z) \cdot B(z)$. □

Uwaga 6.1.1. Wzór na splot

$$[z^n]C(z) = c_n = a_o b_n + a_1 b_{n-1} + a_2 b_{n-2} + ... + a_n b_o = \sum_{k=0}^{n} a_k b_{n-k},\ n \geq 0.$$

□

Przykład 6.1.1. *Rozważmy funkcje tworzące:*

$$\begin{aligned} A(z) &= (1+z)^r = \sum_{k\geq 0} \binom{r}{k} z^k, \\ B(z) &= (1+z)^s = \sum_{k\geq 0} \binom{s}{k} z^k, \end{aligned}$$

wtedy z jednej strony

$$C(z) = (1+z)^{r+s} = \sum_{k\geq 0} \binom{r+s}{k} z^k,$$

a z drugiej, ponieważ $C(z) = A(z)B(z)$, więc ze wzoru na splot:

$$[z^n]C(z) = \sum_{k=0}^{n} \binom{r}{k}\binom{s}{n-k}.$$

Otrzymujemy stąd tożsamość Cauchy'ego:

$$\binom{r+s}{k} = \sum_{k=0}^{n} \binom{r}{k}\binom{s}{n-k}. \qquad \square$$

Ćwiczenie 6.1.1. *Wyznacz splot ciągów $(1-z)^r = [(-1)^k\binom{r}{k}, k = 0, 1, 2, ...]$ oraz $(1+z)^r = [\binom{r}{k}, k = 0, 1, 2, ...]$.* $\square$

Uwaga 6.1.2.

$$\begin{aligned}
(1-z)^r &= [(-1)^k \binom{r}{k}, k = 0, 1, 2, ...], \\
\frac{1}{(1-z)^{n+1}} &= [\binom{k+n}{n}, k = 0, 1, 2, ...], \\
\frac{z^n}{(1-z)^{n+1}} &= [\binom{k}{n}, k = 0, 1, 2, ...].
\end{aligned}$$

Dowód. Jeżeli w (6.0.4) zamiast z wstawimy $-z$ otrzymamy pierwszą tożsamość. Druga tożsamość wynika z pierwszej przy podstawieniu $r = -n - 1$. Rzeczywiście

$$\begin{aligned}
\frac{1}{(1-z)^{n+1}} &= [(-1)^k \binom{-n-1}{k}, k = 0, 1, 2, ...] \\
&= [(-1)^k(-1)^k \binom{k-(-n-1)-1}{k}, k = 0, 1, 2, ...] \\
&= [\binom{k+n}{k}, k = 0, 1, 2, ...] \\
&= [\binom{k+n}{n}, k = 0, 1, 2, ...],
\end{aligned}$$

a trzeci otrzymujemy z drugiego poprzez przemnożenie drugiego przez z^n. Rzeczywiście, jeśli $C(z) = \frac{1}{(1-z)^{n+1}}, D(z) = \frac{z^n}{(1-z)^{n+1}}$ to

$$\begin{aligned}
D(z) &= z^n C(z), \\
[z^k]C(z) &= [z^{k-n}]D(z) = \binom{k}{n}.
\end{aligned} \qquad \square$$

6.2. Rozwiązywanie rekurencji za pomocą funkcji tworzących

Z Uwagi 6.1.2 w szczególności otrzymujemy $\frac{1}{1-z} = [1,1,1,1,1,...]$ a stąd i z definicji splotu, jeśli $A(z) = [a_o, a_1, a_2, ...]$ to $\frac{A(z)}{1-z} = [a_o, a_o+a_1, a_o+a_1+a_2, ...]$ (ciąg sum częściowych).

Zwróćmy też uwagę, że z rozwinięcia funkcji e^z mamy $e^z = [\frac{1}{k!}, k = 0,1,2,...]$.

Podajmy w tabelce niektóre częściej używane funkcje tworzące:

Tabela 6.1. Funkcje tworzące

Ciąg	Suma	Zwarty wzór
$[1,0,0,0,\ldots]$	$\sum_{n\geq 0}[n=0]z^n$	1
$[\underbrace{0,\ldots,0,1}_{m},0,0,\ldots]$	$\sum_{n\geq 0}[n=m]z^n$	z^m
$[1,1,1,1,\ldots]$	$\sum_{n\geq 0} z^n$	$\frac{1}{1-z}$
$[\underbrace{1,\ldots,1}_{m},0,0,\ldots]$	$\sum_{n\geq 0} z^n[n\leq m]$	$\frac{1-z^n}{1-z}$
$[1,-1,1,-1,\ldots]$	$\sum_{n\geq 0}(-1)^n z^n$	$\frac{1}{1+z}$
$[1,0,1,0,\ldots]$	$\sum_{n\geq 0}[2\|n]z^n$	$\frac{1}{1-z^2}$
$[1,\underbrace{0,\ldots,0,1}_{m},\underbrace{0,\ldots,0,1}_{m},\ldots]$	$\sum_{n\geq 0}[m\|n]z^n$	$\frac{1}{1-z^m}$
$[1,2,3,4,\ldots]$	$\sum_{n\geq 0}(n+1)z^n$	$\frac{1}{(1-z)^2}$
$[1,2,4,8,\ldots]$	$\sum_{n\geq 0} 2^n z^n$	$\frac{1}{1-2z}$
$[1^2,2^2,3^2,4^3,5^2,\ldots]$	$\sum_{n=0}(n+1)^2 z^n$	$\frac{1+z}{(1-z)^3}$
$[1,c,\binom{c}{2},\binom{c}{3},\ldots]$	$\sum_{n\geq 0}\binom{c}{n}z^n$	$(1+z)^c$
$[1,c,\binom{c+1}{2},\binom{c+2}{3},\ldots]$	$\sum_{n\geq 0}\binom{c+n-1}{n}z^n$	$\frac{1}{(1-z)^c}$
$[1,c,c^2,c^3,\ldots]$	$\sum_{n\geq 0} c^n z^n$	$\frac{1}{1-cz}$
$[0,c,2c^2,3c^3,4c^4,\ldots]$	$\sum_{n\geq 0} nc^n z^n$	$\frac{cz}{(1-cz)^2}$
$[0,1,\frac{1}{2},\frac{1}{3},\frac{1}{4},\ldots]$	$\sum_{n>0}\frac{1}{n}z^n$	$\ln(\frac{1}{1-z})$
$[0,1,-\frac{1}{2},\frac{1}{3},-\frac{1}{4},\ldots]$	$\sum_{n>0}(-1)^n\frac{1}{n}z^n$	$\ln(1+z)$
$[\frac{1}{0!},\frac{1}{1!},\frac{1}{2!},\frac{1}{3!},\ldots]$	$\sum_{n\geq 0}\frac{1}{n!}z^n$	e^z
$[F_0,F_1,F_2,F_3,F_4,\ldots]$	$\sum_{n\geq 0} F_n z^n$	$\frac{z}{1-z-z^2}$
$[H_0,H_1,H_2,H_3,H_4,\ldots]$	$\sum_{n\geq 0} H_n z^n$	$\frac{1}{1-z}\ln(\frac{1}{1-z})$

gdzie $\{F_n, n \geq 0\}$ są to liczby Fibonacciego (szerzej o nich w dalszej części rozdziału) a $\{H_n, n \geq 1\}$ to ciąg liczb harmonicznych.

Zupełnie elementarnymi metodami można otrzymać następujące zależności, znacznie poszerzające możliwości obliczania współczynników funkcji tworzących i rozwijania funkcji tworzących w szereg.

Twierdzenie 6.2.1. *Jeśli* $A(z) = [a_o, a_1, a_2, ...]$, $B(z) = [b_o, b_1, b_2, ...]$, *to*

Tabela 6.2. Własności funkcji tworzących.

Funkcja generująca	Ciąg
$\alpha A(z) + \beta B(z)$	$[\alpha a_0 + \beta b_0, \alpha a_1 + \beta b_1, \alpha a_2 + \beta b_2, \alpha a_3 + \beta b_3, \ldots]$
$z^m A(z)$	$[\underbrace{0, 0, \ldots, 0}_{m}, a_o, a_1, a_2, a_3, \ldots]$
$\frac{A(z)-(a_o+a_1 z+\ldots+a_{m-1}z^{m-1})}{z^m}$	$[a_m, a_{m+1}, a_{m+2}, a_{m+3}, \ldots]$
$A(\alpha z)$	$[a_0, \alpha a_1, \alpha^2 a_2, \alpha^3 a_3, \ldots]$
$A(z^2)$	$[a_0, 0, a_1, 0, a_2, 0, \ldots]$
$A'(z) = \frac{dA(z)}{dz}$	$[a_1, 2a_2, 3a_3, 4a_4, \ldots]$
$\int_0^z A(t)dt$	$[0, \frac{1}{1}a_0, \frac{1}{2}a_1, \frac{1}{3}a_2, \frac{1}{4}a_3, \ldots]$

□

Przykład 6.2.1. *Rozważmy następującą rekurencję Fibonacciego:*

$$\begin{cases} F_0 &= 0, \\ F_1 &= 1, \\ F_n &= F_{n-1} + F_{n-2} \ \ dla \ \ n \geq 2, \end{cases}$$

Rozwiązanie rekurencji metodą funkcji tworzących składa się z kilku etapów

Etap 1. *Dookreślając, jeśli trzeba, ciąg rekurencyjny zapisz całą rekurencję zwartym wzorem. Tutaj dookreślamy* F_n *dla* $n \in \mathcal{Z}_-$, *przyjmując* $F_n = 0, n \leq 0$. *Ponieważ*

$$\begin{aligned} F_n &= F_{n-1} + F_{n-2}, \ \ n \geq 2 \\ F_1 &= 1 = F_0 + F_{-1} + 1 = F_0 + F_{-1} + [n = 1], \\ F_0 &= 0 = F_{-1} + F_{-2}, \\ F_n &= 0 = F_{n-1} + F_{n-2}, \ \ n < 0, \end{aligned}$$

więc

$$F_n = F_{n-1} + F_{n-2} + [n = 1], \ n \in \mathcal{Z}.$$

Jest to poszukiwany zwarty wzór.

Etap 2. *Pomnóż stronami otrzymaną zwartą rekurencję przez z^n, przyłóż znak sumy po n i przekształć obie strony tak, aby otrzymać funkcję tworzącą nieznanego ciągu rekurencyjnego $F(z) = F_0 + F_1 z + F_2 z^2 + ...$ Mamy więc*

$$F_n z^n = F_{n-1} z^n + F_{n-2} z^n + [n = 1] z^n,\ n \in \mathcal{Z},$$

a stąd

$$\sum_{n\in\mathcal{Z}} F_n z^n = \sum_{n\in\mathcal{Z}} F_{n-1} z^n + \sum_{n\in\mathcal{Z}} F_{n-2} z^n + \sum_{n\in\mathcal{Z}} [n = 1] z^n.$$

czyli

$$\sum_{n\in\mathcal{Z}} F_n z^n = z \sum_{n\in\mathcal{Z}} F_{n-1} z^{n-1} + z^2 \sum_{n\in\mathcal{Z}} F_{n-2} z^{n-2} + \sum_{n\in\mathcal{Z}} [n = 1] z^n.$$

a stąd

$$F(z) = zF(z) + z^2 F(z) + z.$$

Etap 3. *Obliczmy wartość funkcji tworzącej $F(z)$ poszukiwanego ciągu. Z ostatniego równania wyliczamy*

$$F(z) = \frac{z}{1 - z - z^2}.$$

Jest to przedostatni wzór z Tabeli 6.1.

Etap 4. *Znając wzory na różne ciągi funkcji tworzących spróbuj ze zwartej postaci $F(z)$ wyliczyć $F_n = [z^n]F(z), n \geq 0$. Będzie to poszukiwany przez nas ciąg.*

Jak rozwinąć $F(z) = \frac{z}{1-z-z^2}$ w szereg potęgowy. Jest to funkcja wymierna a z analizy wiadomo, że każda funkcja wymierna daje się przedstawić w postaci ułamków prostych. Konkretnie szukamy tutaj przedstawienia

$$\frac{z}{1 - z - z^2} = \frac{A}{1 - \alpha z} + \frac{B}{1 - \beta z}.$$

Zwróćmy uwagę, że rozwinięcia funkcji $\frac{1}{1-cz}$ w szereg potęgowy z Tabeli 6.1 są znane. Sprowadzając prawą stronę do wspólnego mianownika otrzymujemy:

$$\frac{A(1 - \beta z) + B(1 - \alpha z)}{(1 - \alpha z)(1 - \beta z)} = \frac{(A + B) - (A\alpha + B\beta)z}{1 - (\alpha + \beta)z + \alpha\beta z^2} = \frac{z}{1 - z - z^2},$$

co implikuje

$$\begin{cases} A + B & = & 0, \\ A\alpha + B\beta & = & -1, \end{cases}$$

a stąd

$$A = -B = -\frac{1}{\beta - \alpha}.$$

Ponieważ α^{-1} *i* β^{-1} *są pierwiastkami równania* $1 - z - z^2$, *czyli*

$$\begin{cases} \alpha^{-1} = \frac{1+\sqrt{5}}{2}, \\ \beta^{-1} = \frac{1-\sqrt{5}}{2}, \end{cases} \qquad \begin{cases} \alpha = \frac{\sqrt{5}-1}{2}, \\ \beta = -\frac{\sqrt{5}+1}{2}, \end{cases} \qquad \begin{cases} A = \frac{1}{\sqrt{5}}, \\ B = -\frac{1}{\sqrt{5}}, \end{cases}$$

a więc

$$F(z) = \frac{A}{1-\alpha z} + \frac{B}{1-\beta z} = \sum_{n\geq 0} \frac{1}{\sqrt{5}}(\alpha^n - \beta^n)z^n,$$

czyli

$$\boxed{F_n = [z^n]F(z) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right).}$$

□

Opisany powyżej sposób rozwi azywania nie jest może najszybszym sposobem rozwiązania rekurencji Fibonacciego, zależało nam jednak na precyzyjnym opisaniu metodologii rozwiązywania równań rekurencyjnych za pomocą funkcji tworzących.

Liczby Fibonacciego (Leonardo Fibonacci (1175-1250) - włoski matematyk znany również jako: Filius Bonacci (syn Bonacciego) czy Leonardo Pisano (z Pizy)) bardzo często występują w wielu sytuacjach w przyrodzie.

Przykład 6.2.2. Przykłady pojawiania się liczb Fibonacciego.

(i) Jak wybrać dwie liczby naturalne n i m dla których Algorytm Euklidesa (Algorytm 4.1) ma najwięcej obrotów pętli. Okazuje się, że tymi najbardziej "złośliwymi" liczbami s a kolejne liczby Fibonacciego $(F_n, F_{n-1}), n \geq 1$.

(ii) Rozważmy następujący sposób rozmnażania się organizmów:
- osobnik M przekształca sie w następnym pokoleniu w K.
- osobnik K dzieli się w następnym pokoleniu na M i K.
Wtedy zaczynając od osobnika K w n-tym pokoleniu jest F_{n+2} *osobników.*

(iii) (problem Fibonacciego) Pojedyncza para królików jest wprowadzona do środowiska. Po dwóch miesiącach para królików może "wyprodukować" inną parę i później rozmnażanie już trwa co miesiąc. Ile jest par królików po n-miesiącach (oczywiście F_n*).*

(iv) Niech S_n *oznacza ilość tych podzbiorów zbioru* $\{1, 2, 3, ..., n\}$ *które nie zawierają kolejnych elementów, Są to kolejno:* $\emptyset, \{1\}, \{2\}, \{3\}, \{1, 3\}, ...$ *Wtedy* $S_n = F_{n+2}$.

(v) Ułamki łańcuchowe $1 + \frac{1}{1} = \frac{2}{1}$, $1 + \frac{1}{1+\frac{1}{1}} = \frac{3}{2}$, $1 + \frac{1}{1+\frac{1}{1+\frac{1}{1}}} = \frac{5}{3}, ...$ *to ilorazy kolejnych liczb Fibonacciego.* □

Przykład 6.2.3. *Rozwiązać rekurencję:*

$$\begin{cases} g_0 &= 0, \\ g_1 &= 4, \\ g_2 &= 24, \\ g_n &= 6g_{n-1} - 12g_{n-2} + 8g_{n-3}, \quad dla \ n \geq 4, \end{cases}$$

Zgodnie z opisanymi w Przykładzie 6.2.1 etapami postępujemy następująco:

- *Dookreślamy $g_n = 0, n < 0$, a ponieważ*

$$\begin{aligned} g_2 &= 24 = 6g_1 - 12g_0 + 8g_{-1}, \\ g_1 &= 4 = 6g_0 - 12g_{-1} + 8g_{-2} + 4[n = 1], \\ g_0 &= 0 = 6g_{-1} - 12g_{-2} + 8g_{-3}, \end{aligned}$$

stąd otrzymujemy

$$g_n = 6g_{n-1} - 12g_{n-2} + 8g_{n-3} + 4[n = 1], n \in \mathcal{Z}.$$

- *Mnożąc stronami przez z^n mamy*

$$\sum_{n\in\mathcal{Z}} g_n z^n = 6\sum_{n\in\mathcal{Z}} g_{n-1}z^n - 12\sum_{n\in\mathcal{Z}} g_{n-2}z^n + 8\sum_{n\in\mathcal{Z}} g_{n-3}z^n + 4\sum_{n\in\mathcal{Z}} z^n[n = 1].$$

Zatem

$$G(z) = 6zG(z) - 12z^2G(z) + 8z^3G(z) + 4z.$$

- *W końcu otrzymujemy:*

$$G(z) = \frac{4z}{(1-2z)^3}.$$

- *$\star$Zwróćmy teraz uwagę, że $G(z) = z\left(\frac{1}{(1-2z)^2}\right)' = zA'(z)$. Stąd $g_n = na_n, n \in \mathcal{Z}$. Zauważmy, że $A(z) = \left(\frac{1}{1-2z}\right)^2$ a ponieważ $\frac{1}{1-2z} = [1, 2, 2^2, 2^3, ...]$, zatem ze wzoru na splot*

$$[z^n]A(z) = \sum_{k=0}^{n} 2^k \cdot 2^{n-k} = (n+1)2^n, n \geq 0,$$

a więc

$$[z^n]G(z) = n(n+1)2^n, n \geq 0.$$

$\star$ Inny sposób. Mamy

$$[z^n]\frac{1}{(1-z)^3} = [1, 3, \binom{4}{2}, \binom{5}{3}, ...] = [\frac{(n+1)(n+2)}{2}, n \geq 0].$$

Ponadto

$$[z^n]\frac{1}{(1-2z)^3} = [\frac{(n+1)(n+2)}{2}2^n, n \geq 0] = [(n+1)(n+2)2^{n-1}, n \geq 0],$$

a stąd

$$[z^n]\frac{4z}{(1-2z)^3} = [n(n+1)2^n, n \geq 0].$$

Jednak początkowe dwa kroki można wykonać trochę inaczej. Przedstawimy tutaj ten sposób postępowania.

◇ *Mnożymy bezpośrenio równania określające rekurencję (pierwsze przez z^0, drugie przez z^1, trzecie przez z^2 a czwarte przez z^n)*

$$\begin{cases} g_0 = 0, & / \cdot z^0 \\ g_1 = 4, & / \cdot z^1 \\ g_2 = 24, & / \cdot z^2 \\ g_n = 6g_{n-1} - 12g_{n-2} + 8g_{n-3}, & / \cdot z^n \\ & dla\ n \geq 4, \end{cases}$$

otrzymując

$$\begin{cases} g_0z^0 = 0, \\ g_1z^1 = 4z^1, \\ g_2z^2 = 24z^2, \\ g_3z^3 = 6g_2z^3 - 12g_1z^3 + 8g_0z^3, \\ g_4z^4 = 6g_3z^4 - 12g_2z^4 + 8g_1z^4, \\ g_5z^5 = 6g_4z^5 - 12g_3z^5 + 8g_2z^5, \\ g_6z^6 = 6g_5z^6 - 12g_4z^6 + 8g_3z^6, \\ \vdots \quad\quad \vdots \\ g_nz^n = 6g_{n-1}z^n - 12g_{n-2}z^n + 8g_{n-3}z^n, \\ \vdots \quad\quad \vdots \end{cases}$$

Sumujemy stronami

$$\begin{aligned} g_0z_0 + g_1z_1 + \ldots &= 6z(g_2z^2 + g_3z^3 + \ldots) - 12z^2(g_1z^1 + g_2z^2 + \ldots) \\ &\quad +8z^3(g_oz^0 + g_1z^1 + \ldots) + 4z + 24z^2, \\ G(z) &= 6z(G(z) - g_1z^1 - g_0z^0) - 12z^2(G(z) - g_0z^0) \\ &\quad +8z^3G(z) + 4z + 24z^2, \\ G(z) &= G(z)(6z - 12z^2 + 8z^3) - 24z^2 + 4z + 24z^2, \\ G(z)(1 - 6z + 12z^2 - 8z^3) &= 4z, \\ G(z) &= \frac{4z}{(1-2z)^3}. \end{aligned}$$

Otrzymaliśmy więc identyczne równanie jak w pierwszym sposobie. W pozostałych punktach postępujemy jak powyżej. □

6.3. Algebra wielomianów rzeczywistych

Przypomnijmy najpierw kilka podstawowych faktów dotyczących operacji na wielomianach. Niech $P(x)$ będzie wielomianem o współczynnikach całkowitych. Stopniem wielomianu $P(x) = p_0 + p_1 x + \ldots + p_n x^n$, nazywamy liczbę $\deg P(x) = n$, równą najwyższej potędze x w tym wielomianie a współczynnik przy najwyższej potędze n (tutaj p_n) nazywać będziemy najwyższym współczynnikiem. Stopień wielomianu jest to najmniejsza liczba naturalna n taka, że $\triangle^n P(x) \equiv 0$.

6.3.1. Podzielność wielomianów

Wiele operacji na wielomianach ma swoje analogie w operacjach na liczbach całkowitych. Na przykład odpowiednikiem wzoru (3.4.1), leżącego u podstaw algorytmu Euklidesa, jest następujący wynik:

Twierdzenie 6.3.1. *Niech $A(x)$ oraz $B(x)$ będą wielomianami $0 < \deg B(x)$. Wtedy istnieje dokładnie jeden układ wielomianów $Q(x)$ oraz $R(x)$ takich, że*

$$A(x) = Q(x)B(x) + R(x),$$

gdzie $\deg R(x) < \deg B(x)$. Wielomian $Q(x)$ nazywamy ilorazem a $R(x)$ resztą.

Dowód. Jeśli $\deg A(x) < \deg B(x)$ to wystarczy przyjąć $Q(x) = 0, R(x) = A(x)$. Załóżmy, że twierdzenie jest prawdziwe dla wielomianów $A(x)$ stopnia $< n$ i niech $\deg A(x) = n \geq \deg B(x) = q$. Przez a oznaczmy najwyższy współczynnik wielomianu $A(x)$ (współczynnik przy najwyższej potędze x) a przez b najwyższy współczynnik wielomianu $B(x)$. Ponieważ stopień wielomianu $ax^{n-q}B(x)/b$ jest równy n a najwyższy współczynnik jest równy a więc $A_1(x) = A(x) - \frac{a}{b}x^{n-q}B(x)$ jest wielomianem stopnia $\deg A_1(x) < n$, z założenia indukcyjnego istnieją więc wielomiany $Q_1(x)$ i $R(x)$ takie, że $A_1(x) = Q_1(x)B(x) + R(x)$ a stąd

$$A(x) = (\frac{a}{b}x^{n-q} + Q_1(x))B(x) + R(x),$$

jest szukanym rozkładem.

Wykażemy teraz jednoznaczność rozkładu. Jeśli $A(x) = Q(x)B(x) + R(x)$ i $A(x) = Q_1(x)B(x) + R_1(x)$, to $(Q_1(x) - Q(x))B(x) = R(x) - R_1(x)$, a ponieważ $\deg(R(x) - R_1(x)) < \deg B(x)$ więc $Q_1 \equiv Q$ a stąd $R_1 \equiv R$. □

Konstrukcyjny dowód tego twierdzenia umożliwia opracowanie algorytmu do obliczania Q i R. Algorytm ten jest znanym procesem dzielenia wielomianów:

Przykład 6.3.1. *Znaleźć wielomiany $Q(x)$ i $R(x)$ w rozwinięciu z Twierdzenia 6.3.1, dla*

$$A(x) = 3x^5 + 5x^4 + 2x^3 + x^2 + 2 \quad \textit{oraz} \quad B(x) = x^3 + 2x^2 - 1.$$

Wykonujemy dzielenie wielomianów

$$\begin{array}{rl}
 & 3x^2 - x + 4 \\
\hline
 & \left(3x^5 + 5x^4 + 2x^3 + x^2 + 2\right) : \left(x^3 + 2x^2 - 1\right) \\
- & 3x^5 + 6x^4 - 3x^2 \\
\hline
 & -x^4 + 2x^3 + 4x^2 \\
- & -x^4 - 2x^3 + x \\
\hline
 & 4x^3 + 4x^2 - x + 2 \\
- & 4x^3 + 8x^2 - 4 \\
\hline
 & -4x^2 - x + 6
\end{array}$$

i otrzymujemy

$$Q(x) = 3x^2 - x + 4, \quad \textit{oraz} \quad R(x) = -4x^2 - x + 6.$$

Ponadto $\deg A(x) = 5 = 2 + 3 = \deg Q(x) + \deg B(x)$. □

Definicja 6.3.1. *Mówimy, że wielomian $B(x)$ jest dzielnikiem wielomianu $A(x)$ i zapisujemy ten fakt $B|A$ jeśli w rozkładzie z Twierdzenia 6.3.1 zachodzi $R(x) \equiv 0$.* □

Definicja 6.3.2. *Wielomian $C(x)$ nazywamy największym wspólnym dzielnikiem wielomianów $A(x)$ i $B(x)$ (zapisujemy $NWD(A, B)$) jeśli:*
(i) Najwyższym współczynnikiem wielomianu C jest 1.
(ii) $C|A$ oraz $C|B$.
(iii) Jeśli $D|A$ i $D|B$ to $D|C$. □

Największy wspólny dzielnik dwóch wielomianów może być wyznaczony algorytmem analogicznym do Algorytmu 4.1 (Algorytm Euklidesa) gdzie operację dzielenia i obliczania reszty określone są w Twierdzeniu 6.3.1 i Przykładzie 6.3.1.

Przykład 6.3.2. *Znajdź $NWD(P, Q)$ gdzie*

$$\begin{aligned}
P(x) &= x^7 - 14 \cdot x^6 + 7 \cdot x^5 + 35 \cdot x^4 - 17 \cdot x^3 + 43 \cdot x^2 - 23 \cdot x - 6 \\
Q(x) &= x^5 - 14 \cdot x^4 + 8 \cdot x^3 + 20 \cdot x^2 + 7 \cdot x + 34
\end{aligned}$$

Mamy

$$\begin{aligned} P(x) &= Q(x)(x^2-1)+(x^4-16x^3+29x^2-16x+28), \\ Q(x) &= (x^4-16x^3+29x^2-16x+28)(x+2)+(11x^3-22x^2+11x-22), \\ R(x) &= (11x^3-22x^2+11x-22)(\frac{x}{11}-\frac{14}{11})+0, \\ & \quad \textit{gdzie } R(x)=x^4-16x^3+29x^2-16x+28, \end{aligned}$$

a więc $NWD(P,Q)(x)=x^3-2x^2+x-2=(x^2+1)(x-2)$. □

Definicja 6.3.3. *Wielomiany $A(x)$ i $B(x)$ nazywamy względnie pierwszymi jeżeli $NWD(A,B)=1$.* □

Analogicznie do Uwagi 4.1.4 możemy podać

Twierdzenie 6.3.2. *Dla każdych dwóch wielomianów $A(x)$ i $B(x)$ istnieją wielomiany $P(x)$ i $Q(x)$ takie, że*

$$A(x)P(x)+B(x)Q(x)=NWD(A,B)(x). \qquad \square$$

Dowód przebiega tak jak dowód Uwagi 4.1.4, natomiast wyznaczenie wielomianów $P(x)$ i $Q(x)$ jest analogiczne do rozszerzonego algorytmu Euklidesa (zob. Algorytm 4.2).

Przykład 6.3.3. *Przenalizujmy Przykład 6.3.2. Niech wielomiany $P(x)$ i $Q(x)$ będą takie, jak w przykładzie 6.3.2. Oznaczmy:*

$$A(x)=x^2-1, \quad B(x)=x-14, \quad C(x)=x+2, \quad D(x)=11x^3-22x^2+11x-22,$$

wtedy

$$\begin{aligned} P(x) &= Q(x)A(x)+B(x), \\ Q(x) &= B(x)C(x)+D(x), \\ B(x) &= D(x)E(x), \end{aligned}$$

a stąd

$$\begin{aligned} D(x) &= Q(x)-B(x)C(x) \\ &= Q(x)-(P(x)-A(x)Q(x))C(x) \\ &= P(x)(-C(x))+Q(x)(1+A(x)C(x)) \\ &= P(x)(-x-2)+Q(x)(x^3+2x^2-x-1) \end{aligned}$$

zatem

$$NWD(P,Q)(x) = \frac{D(x)}{11} = P(x)(-\frac{1}{11}x-\frac{2}{11})+Q(x)(\frac{1}{11}x^3+\frac{2}{11}x^2-\frac{1}{11}x-\frac{1}{11}).$$

□

Bezpośrednio z Twierdzenia 6.3.2 otrzymujemy:

Twierdzenie 6.3.3. *Na to aby wielomiany $A(x)$ i $B(x)$ były względnie pierwsze potrzeba i wystarcza aby istniały wielomiany $P(x)$ i $Q(x)$ takie, że*

$$A(x)P(x) + B(x)Q(x) \equiv 1.$$ □

6.3.2. Zasadnicze twierdzenie algebry

Analogicznie do liczb pierwszych w zbiorze liczb naturalnych (zob. podrozdz. 4.2) wprowadzamy wielomiany pierwsze:

Definicja 6.3.4. *Mówimy, że wielomian $P(x)$ stopnia* $\deg P(x) > 0$ *jest pierwszy jeśli jest wielomianem unormowanym (monicznym) , to znaczy jego najwyższy współczynnik wynosi 1 oraz nie istnieją wielomiany stopni dodatnich $A(x)$ i $B(x)$ takie, że $P(x) = A(x)B(x)$.* □

Postać wielomianów pierwszych zależy istotnie od tego z jakiej klasy brane są współczynniki wielomianów. Na przykład dla wielomianów o współczynnikach zespolonych wielomian $x^2 + 1$ jest złożony, bo

$$x^2 + 1 = (x + \sqrt{-1})(x - \sqrt{-1}),$$

natomiast dla wielomianów o współczynnikach rzeczywistych ten wielomian jest pierwszy. Gdyby nie był to istniałby rozkład dla liczb rzeczywistych a, b, c taki, że

$$x^2 + 1 = a(x - b)(x - c),$$

co jak wiemy jest niemożliwe, bo wyróżnik trójmianu $x^2 + 1$ jest ujemny. Nas w tym rozdziale interesować będą wielomiany o współczynnikach rzeczywistych. Wtedy wielomiany pierwsze mają ściśle określoną postać:

Twierdzenie 6.3.4. *Wśród wielomianów o współczynnikach rzeczywistych jedynymi pierwszymi wielomianami są wielomiany stopnia pierwszego (tzn. postaci $x - a, a \in \Re$) oraz wielomiany postaci $x^2 + px + q$, gdzie $p^2 - 4q < 0$.*
□

Analogicznie do Twierdzenia 4.2.1 (Podstawowego Twierdzenia Arytmetyki) i Twierdzenia 4.2.2 mamy

Twierdzenie 6.3.5. Podstawowe twierdzenie algebry wielomianów rzeczywistych *Każdy wielomian rzeczywisty $A(x)$ można przedstawić w postaci iloczynu stałej (jest to najwyższy współczynnik wielomianu) i wielomianów pierwszych. Jeśli mamy dwa takie przedstawienia $A(x) = a_n \prod_{i=1}^{m} P_i(x) = a_n \prod_{i=1}^{m} Q_i(x)$ to różnią się one tylko porządkiem, to znaczy ciąg wielomianów $\{P_i, 1 \leq i \leq m\}$ jest permutacją $\{Q_i, 1 \leq i \leq m\}$.)* □

Stąd i z Twierdzenia 6.3.4 otrzymujemy

Twierdzenie 6.3.6. *Dla każdego wielomianu o współczynnikach rzeczywistych $A(x)$ istnieją liczby rzeczywiste $a_o, p_i, q_i, 1 \leq i \leq l, x_i l + 1 \leq i \leq k$ oraz naturalne $s_i, 1 \leq i \leq k$, takie, że $p_i^2 - 4q_i < 0, 1 \leq i \leq l$, oraz*

$$A(x) = a_0(x^2 + p_1 x + q_1)^{s_1} \dots (x^2 + p_l x + q_l)^{s_l}(x - x_{l+1})^{s_{l+1}} \dots (x - x_k)^{s_k}, \tag{6.3.2}$$

gdzie $\deg A(x) = 2\sum_{i=1}^{l} s_i + \sum_{i=l+1}^{k} s_i$. □

Znajdowanie rozkładu (6.3.2) dla dowolnego wielomianu (porównaj Algorytm faktoryzacji 4.3) może być czasem bardzo trudne. Pomocne w takiej sytuacji jest następujące twierdzenie

Twierdzenie 6.3.7. (Bézout É.) *Resztą z dzielenia wielomianu $P(x)$ przez dwumian $(x - a)$ jest $P(a)$.* □

Twierdzenie 6.3.8. *Jeżeli liczba wymierna $\frac{p}{q}$ $(p \perp q)$ jest miejscem zerowym wielomianu $A(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_0$ o współczynnikach całkowitych $(a_i \in \mathcal{Z}, 0 \leq i \leq n)$ to $p|a_0$ oraz $q|a_n$.*

Dowód. Ponieważ $A(\frac{p}{q}) = 0$ więc

$$a_n p^n + a_{n-1}p^{n-1}q + a_{n-2}p^{n-2}q^2 + \ldots + a_1 pq^{n-1} + a_0 q^n = 0,$$

i uwzględniając fakt, że $p \perp q$ z podzielności tych czynników otrzymujemy tezę. □

Twierdzenie 6.3.8 wskazuje, że aby znaleźć miejsca zerowe wielomianu o współczynnikach całkowitych należy przebadać dzielniki wyrazu wolnego.

Przykład 6.3.4. *Rozłóż wielomian $A(x) = 2x^4 + x^3 - 8x^2 + x - 10$ na czynniki pierwsze. Ponieważ $10 = 1 \cdot 2 \cdot 5$ więc "podejrzane" liczby to $p =$*

$1, 2, 5, 10, -1, -2, -5, -10,$ *a* $q = 1, 2$

p	q	$\frac{p}{q}$	$A(\frac{p}{q})$
1	1	1	-14
2	1	2	0
5	1	5	1170
10	1	10	40400
-1	1	-1	-18
-2	1	-2	-20
-5	1	-5	1210
-10	1	-10	18180

p	q	$\frac{p}{q}$	$A(\frac{p}{q})$
1	2	$\frac{1}{2}$	$-11\frac{1}{4}$
2	2	1	-14
5	2	$2\frac{1}{2}$	$36\frac{1}{4}$
10	2	5	1170
-1	2	$-\frac{1}{2}$	$-6\frac{1}{4}$
-2	2	-1	-18
-5	2	$-2\frac{1}{2}$	0
-10	2	-5	1210

a więc $A(x)$ *dzieli się przez* $(x-2)$ *oraz* $(x+2\frac{1}{2})$*. Dzieląc, jak w przykładzie 6.3.1,* $A(x)$ *przez* $(x-2)$ *otrzymujemy* $B(x) = 2x^3+5x^2+2x$*, a dalej dzieląc* $B(x)$ *przez* $(x+2\frac{1}{2})$ *otrzymujemy* $C(x) = 2x^2+2 = 2(x^2+1)$*, a ponieważ* $\triangle = 0 - 4 = -4$ *jest ujemna, więc wielomian* x^2+1 *jest nierozkładalny. Stąd*

$$A(x) = 2(x^2+1)(x+2\frac{1}{2})(x-2),$$

i jest to poszukiwany rozkład na czynniki pierwsze. □

6.3.3. Funkcje wymierne

Definicja 6.3.5. *Funkcja wymierna* $R(x)$ *jest to funkcja postaci* $\frac{P(x)}{Q(x)}$*, gdzie* $P(x)$ *oraz* $Q(x) \neq 0$ *są wielomianami skończonego stopnia takie, że* $NWD(P,Q) \equiv 1$*. Przedstawienie funkcji wymiernej jest jednoznaczne z dokładnością do stałej.* □

Definicja 6.3.6. *Funkcję wymierną* $R(x)$ *nazywamy ułamkiem prostym jeśli ma ona postać*

$$R(x) = \frac{Q(x)}{(P(x))^n},$$

gdzie P *jest wielomianem pierwszym i* $\deg Q(x) < \deg P(x)$*.* □

Mimo licznych analogii wielomianów do liczb naturalnych, wielomiany są funkcjami ciągłymi i pewne wyniki są związane z zachowaniem funkcji. Następujące twierdzenie jest interesującym uogólnieniem wzoru Taylora:

Twierdzenie 6.3.9. *Jeżeli* $A(x)$ *i* $B(x)$ *są dowolnymi wielomianami i* $\deg B(x) > 0$ *to istnieją wielomiany* $C_i(x), 0 \leq i \leq q = \lfloor \frac{\deg A(x)}{\deg B(x)} \rfloor$*, takie że* $\deg C_i(x) < \deg B(x), 0 \leq i \leq q$*, oraz*

$$A(x) = C_0(x)B^q(x) + C_1(x)B^{q-1}(x) + \ldots C_{q-1}(x)B(x) + C_q(x).$$

Dowód. Załóżmy, że twierdzenie jest prawdziwe dla wielomianów $A(x)$ stopnia $< l$ i niech $\deg A(x) = l$. Z twierdzenia 6.3.1 istnieją wielomiany $C_0(x)$ i $A_1(x)$ takie, że

$$A(x) = C_0(x)B^q(x) + A_1(x),$$

i $\deg A_1(x) < \deg B^q(x) \leq \deg A(x)$. Ponadto musimy mieć $\deg C_0(x) < \deg B(x)$ w przeciwnym razie liczba q byłaby źle zdefiniowana. Teraz stosując założenie indukcyjne do wielomianów $A_1(x)$ i $B(x)$ otrzymujemy tezę. □

Przyjmując $B(x) = x - a$ można trochę więcej w tym przypadku powiedzieć o wielomianach $C_i(x)$. Wtedy $C_i(x) = \frac{d^i}{(dx)^i}A(a) = A^{(i)}(a)$, a rozwinięcie staje się rozwinięciem Taylora.

Twierdzenie 6.3.10. *Każda funkcja wymierna jest sumą wielomianu i ułamków prostych.*

Dowód. Niech $R(x) = \frac{A(x)}{B(x)}$, gdzie $B(x) = \prod_{i=1}^{n} P_i^{l_i}(x)$ jest rozkładem wielomianu $B(x)$ na czynniki pierwsze (stał a przy rozwinięciu przyjmyjemy 1). Dowodzimy indukcyjnym względem n. Jeśli $n = 1$ to $R(x) = \frac{A(x)}{P_1^{l_1}(x)}$ i stosując teraz Twierdzenia 6.3.9 z $B(x)$ zastąpionym przez $P_1(x)$ otrzymujemy tezę.

Jeśli twierdzenie jest prawdziwe dla $n-1$ to ponieważ $NWD(\prod_{i=1}^{n-1} P_i^{l_i}, P_n^{l_n})(x) = 1$, więc z Twierdzenia 6.3.3 istnieją wielomiany $K_1(x)$ i $K_2(x)$ takie, że

$$K_1(x)\prod_{i=1}^{n-1} P_i^{l_i}(x) + K_2(x)P_n^{l_n}(x) = 1,$$

a stąd

$$R(x) = \frac{A(x)}{\prod_{i=1}^{n} P_i^{l_i}(x)} = \frac{A(x)K_1(x)}{P_n^{l_n}(x)} + \frac{A(x)K_2(x)}{\prod_{i=1}^{n-1} P_i^{l_i}(x)}.$$

Z Twierdzenia 6.3.9 pierwszy składnik jest sumą wielomianu i ułamków prostych a drugi jest taką sumą z założenia indukcyjnego. □

Przykład 6.3.5. *Opisaną wyżej metodę ogólną zilustrujemy na przykładzie funkcji*

$$R(x) = \frac{x^4 - x^2 + 1}{x^3 - x^2 - x + 1}.$$

Dzieląc, tak jak w przykładzie 6.3.1, $x^4 - x^2 + 1$ przez $x^3 - x^2 - x + 1$ otrzymujemy rozkład:

$$R(x) = (x + 1) + \frac{x^2}{x^3 - x^2 - x + 1}.$$

Spróbujmy teraz rozłożyć $Q(x) = x^3 - x^2 - x + 1$ *na czynniki pierwsze. Korzystając z Twierdzeń 6.3.7 i 6.3.8 widzimy, że "podejrzanymi" punktami są* $-1, 1$ *i rzeczywiście* $Q(-1) = Q(1) = 0$. *Dzieląc teraz wielomian* $Q(x)$ *przez* $(x+1)$ *otrzymujemy* $\frac{Q(x)}{x+1} = x^2 - 2x + 1$ *i ze wzorów skróconego mnożenia lub po podzieleniu przez* $(x-1)$ *widzimy, że*

$$x^3 - x^2 - x + 1 = (x-1)^2(x+1),$$

a stąd i z Twierdzenia 6.3.10 (z dowodu) otrzymujemy rozkład

$$\frac{x^2}{x^3 - x^2 - x + 1} = \frac{x^2}{(1-x)^2 \cdot (1+x)} = \frac{\alpha}{1-x} + \frac{\beta}{(1-x)^2} + \frac{\gamma}{1+x}.$$

Mnożąc obie strony przez $(1-x)^2 \cdot (1+x)$ *otrzymujemy:*

$$x^2 = \alpha\left(1 - x^2\right) + \beta\,(1+x) + \gamma\left(1 - 2x + x^2\right).$$

Dwa wielomiany są równe, gdy współczynniki przy odpowiadających potęgach są sobie równe. Wartości α, β, γ *można więc wyliczyć z układu równań*

$$\begin{cases} \alpha + \beta + \gamma = 0 \\ \beta - 2\gamma = 0 \\ -\alpha + \gamma = 1. \end{cases}$$

Rozwiązaniem powyższego układu są wartości $\alpha = -\frac{3}{4},\ \beta = \frac{1}{2},\ \gamma = \frac{1}{4}$. *W konsekwencji ze wzorów z Tabeli 6.1 otrzymujemy szereg*

$$\begin{aligned} R(x) &= (x+1) + \sum_{n=0}^{\infty}\left(-\frac{3}{4} + \frac{1}{2}\,(n+1) + \frac{1}{4}\,(-1)^n\right)x^n \\ &= 1 + x + x^2 + x^3 + 2x^4 + 2x^5 + 3x^6 + 3x^7 + 4x^8 + \ldots. \end{aligned}$$ □

Przykład 6.3.6. *Rozwiń funkcję tworzącą* $R(z) = \frac{5z^4 - 2z^3 + 4z^2 + z - 2}{3z^5 - z^4 + 6z^3 - 2z^2 + 3z - 1}$, *w szereg. W tym celu rozkładamy mianownik na czynniki pierwsze. Z Twierdzeń 6.3.7 i 6.3.8 "podejrzanymi" punktami są* $\frac{-1}{3}, \frac{1}{3}, 1, -1$ *a wartości mianownika w tych punktach to odpowiednio* $2\frac{38}{81}, 0, 16, -8$ *a więc dwumian* $(z - \frac{1}{3})$ *jest dzielnikiem mianownika. Stąd*

$$3z^5 - z^4 + 6z^3 - 2z^2 + 3z - 1 = 3(z - \frac{1}{3})(1 + 2z^2 + z^4).$$

Dalej, ze wzorów skróconego mnożenia, $1 + 2z^2 + z^4 = (1 + z^2)^2$ *więc rozkład na czynniki pierwsze mianownika jest następujący:*

$$3z^5 - z^4 + 6z^3 - 2z^2 + 3z - 1 = 3(z - \frac{1}{3})(1 + z^2)^2.$$

Stąd

$$R(z) = \frac{az+b}{1+z^2} + \frac{cz+d}{(1+z^2)^2} + \frac{e}{3z-1}.$$

Zatem

$$\begin{aligned} & 5z^4 - 2z^3 + 4z^2 + z - 2 \\ = \; & (az+b)(1+z^2)(3z-1) + (cz+d)(3z-1) + e(1+z^2)^2, \\ = \; & (3a+e)z^4 + (3b-a)z^3 \\ & +(3a+3c+2e-b)z^2 + (3b-a+3d-c)z + (e-b-d), \end{aligned}$$

czyli współczynniki a, b, c, d, e *spełniają układ równań*

$$\left\{ \begin{array}{rrrrrcl} 3a & & & & +e & = & 5, \\ -a & +3b & & & & = & -2, \\ 3a & -b & +3c & & +2e & = & 4, \\ -a & +3b & -c & +3d & & = & 1, \\ & -b & & -d & +e & = & -2. \end{array} \right.$$

Stąd $a = 2, b = c = 0, d = 1, e = -1$ *a więc*

$$R(z) = \frac{1}{(1+z^2)^2} + \frac{2z}{1+z^2} - \frac{1}{1-3z}.$$

Teraz wykorzystujemy tabele 6.1 i 6.2. Ponieważ $\frac{1}{1+z} = [(-1)^n, n \geq 0]$ *więc* $\frac{1}{1+z^2} = [(-1)[n \bmod 4 = 2] + 1[n \bmod 4 = 0], n \geq 0]$ *i ze wzoru na splot (Uwaga 6.1.1)* $\frac{1}{(1+z^2)^2} = [\sum_{k=0}^{n}((-1)[k \bmod 4 = 2] + [k \bmod 4 = 0])((-1)[n-k \bmod 4 = 2] + [n-k \bmod 4 = 0]), n \geq 0]$ *oraz* $\frac{z}{1+z^2} = [(-1)[n-1 \bmod 4 = 2] + [n-1 \bmod 4 = 0], n \geq 0]$ *a stąd i z* $\frac{-1}{1-3z} = [-3^n, n \geq 0]$

$$\begin{aligned} [z^n]R(z) &= \sum_{k=0}^{n}([k \bmod 4 = 2, n \bmod 4 = 0] + [k \bmod 4 = 0, n \bmod 4 = 0]) \\ &\quad +(-1)([k \bmod 4 = n \bmod 4 = 2] + [k \bmod 4 = 0, n \bmod 4 = 2])) \\ &\quad +(-2)[n-1 \bmod 4 = 2] + 2[n-1 \bmod 4 = 0] - 3^n, \\ &= \left\{ \begin{array}{ll} \frac{n}{2} - 3^n, & \textit{jeżeli } n \bmod 4 = 0, \\ 2 - 3^n, & \textit{jeżeli } n \bmod 4 = 1, \\ -\frac{n}{2} - 3^n, & \textit{jeżeli } n \bmod 4 = 2, \\ -2 - 3^n, & \textit{jeżeli } n \bmod 4 = 3. \end{array} \right. \end{aligned}$$

□

Jeżeli mianownik $Q(x)$ funkcji wymiernej $R(x) = \frac{P(x)}{Q(x)}$ posiada jedynie pierwiastki jednokrotne, to następne twierdzenie znacznie przyspiesza rozkład $R(x)$ w szereg potęgowy.

Twierdzenie 6.3.11. *Jeśli* $R(x) = P(x)/Q(x)$, *gdzie* $Q(x) = q_0 \cdot (1 - \rho_1 x) \cdot \ldots \cdot (1 - \rho_l x)$ *i liczby* $\rho_1, \ldots, \rho_l$ *są parami różne, to w przypadku gdy* $P(x)$ *jest wielomianem stopnia mniejszego niż* l, *zachodzi*

$$[x^n]R(x) = a_1\rho_1^n + \ldots + a_l\rho_l^n, \quad dla\, a_k = \frac{-\rho_k \cdot P(1/\rho_k)}{Q'(1/\rho_k)} = \frac{P(1/\rho_k)}{\rho_0 \prod_{j\neq k}(1 - \rho_j/\rho_k)}.$$

□

Przykład 6.3.7. *Mianownik* $Q(x)$ *funkcji wymiernej*

$$R(x) = \frac{P(x)}{Q(x)} = \frac{2x}{1 - 5x - 2x^2 + 24x^3}.$$

ma trzy różne pierwiastki i można $R(x)$ *przedstawić jako*

$$R(x) = \frac{2x}{(1 + 2x)\,(1 - 3x)\,(1 - 4x)}.$$

W Twierdzenia 6.3.11 otrzymujemy więc

k	ρ_k	$P(1/\rho_k)$	$\prod_{j\neq k}(1 - \rho_j/\rho_k)$	a_k
0	1			
1	-2	-1	$\frac{15}{2}$	$-\frac{2}{15}$
2	3	$\frac{2}{3}$	$-\frac{5}{9}$	$-\frac{6}{5}$
3	4	$\frac{1}{2}$	$\frac{3}{8}$	$\frac{4}{3}$

tak, że

$$[x^n]R(x) = -\frac{2}{15}\,(-2)^n - \frac{6}{5}3^n + \frac{4}{3}4^n.$$

□

Przykład 6.3.8. *Z Twierdzenia 6.3.11 rekurencję Fibonacciego możemy rozwiązać prawie natychmiast. Teraz* $P(z) = z, Q(z) = (1 - \frac{1+\sqrt{5}}{2}z)(1 - \frac{1-\sqrt{5}}{2}z)$, *a stąd* $F_1 = \frac{\frac{2}{1+\sqrt{5}}}{(1-1)!(1-\frac{1-\sqrt{5}}{1+\sqrt{5}})} = \frac{1}{\sqrt{5}}$ *i* $F_2 = \frac{\frac{2}{1-\sqrt{5}}}{(1-1)!(1-\frac{1+\sqrt{5}}{1-\sqrt{5}})} = -\frac{1}{\sqrt{5}}$ *a więc*

$$F_n = [z^n]F(z) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right).$$

□

Przykład 6.3.9. *Niech* $a_n = (1 + \sqrt{2})^n + (1 - \sqrt{2})^n, n \geq 0$. *Znajdź proste równanie rekurencyjne, które spełniają te współczynniki.*

Rozwiązanie:

Z Twierdzenia 6.3.11 możemy wywnioskować, że funkcja tworząca tego ciągu ma postać

$$\begin{aligned} A(z) &= \frac{1}{1-(1+\sqrt{2})z} + \frac{1}{1-(1-\sqrt{2})z} \\ &= \frac{2-2z}{1-2z-z^2} \end{aligned}$$

a stąd

$$A(z) - 2A(z)z - z^2A(z) = 2 - 2z,$$

czyli

$$A(z) = 2zA(z) + z^2A(z) + 2 - 2z,$$

więc

$$\begin{cases} a_o = 2, \ \ a_1 = 2, \\ a_n = 2a_{n-1} + a_{n-2}, \ n \geq 2. \end{cases}$$

□

Twierdzenie 6.3.11 możemy stosować dla jednokrotnych pierwiastków mianownika $Q(x)$, następujący wynik wyjaśnia co się dzieje gdy mamy wielokrotne pierwiastki rzeczywiste:

Twierdzenie 6.3.12. O rozwijaniu wymiernych funkcji tworzących.

Jeśli $R(z) = P(z)/Q(z)$, gdzie $Q(z) = (1-\rho_1 z)^{d_1}...(1-\rho_l z)^{d_l}$ i liczby $\{\rho_1, \rho_2, ..., \rho_l\}$ są parami różne, to w przypadku gdy $P(z)$ jest wielomianem stopnia mniejszego niż $d_1 + d_2 + ... + d_l$, zachodzi

$$[z^n]R(z) = f_1(n)\rho_1^n + \ldots + f_l(n)\rho_l^n,$$

dla wszystkich $n \geq 0$ gdzie wielomian $f_k(n)$ jest wielomianem stopnia $d_k - 1$ z najbardziej znaczącym współczynnikiem

$$a_k = \frac{P(1/\rho_k)}{(d_k-1)!\prod_{j\neq k}(1-\rho_j/\rho_k)^{d_j}}, \ \ 1 \leq k \leq l.$$

□

Przykład 6.3.10. *Rozwiązać rekurencję*

$$\begin{cases} r_0 = r_1 = r_2 = r_3 = 1, \ r_4 = 2, \\ r_n = r_{n-1} + r_{n-2} - r_{n-3}, n \geq 5. \end{cases}$$

Postępując jak w Przykładzie 6.2.1 określamy rekurencję zwartym wzorem:

$$r_n = r_{n-1} + r_{n-2} - r_{n-3} + [n=4] - [n=2] + [n=0], n \in \mathcal{Z},$$

której funkcja tworząca spełnia równanie

$$R(z) = \frac{z^4 - z^2 + 1}{1 - z - z^2 + z^3} = (z+1) + \frac{z^2}{(1-z)^2(1+z)}.$$

Stosujemy Twierdzenie 6.3.12 z $P(x) = x^2$ *oraz* $Q(x) = (1-x)^2(1+x)$ *a więc*

k	ρ_k	d_k	$P(1/\rho_k)$	$(d_k-1)!\prod_{j\neq k}(1-\rho_j/\rho_k)^{d_j}$	a_k
0	1				
1	1	2	1	2	$\frac{1}{2}$
2	-1	1	1	4	$\frac{1}{4}$

stąd

$$R(z) = 1 + z + \sum_{n=0}^{\infty}(\frac{1}{2}n + \alpha + \frac{1}{4}(-1)^n)z^n,$$

a ponieważ

$$1 = r_1 = 1 + \frac{1}{2} + \alpha - \frac{1}{4},$$

(jest to współczynnik przy z) stąd $\alpha = -\frac{1}{4}$. *Zwróćmy uwagę, że rozważana tutaj funkcja tworząca jest identyczna jak ta w Przykładzie 6.3.5.* □

Ćwiczenie 6.3.1. *Rozwiąż rekurencję*

$$\begin{cases} g_0 &= g_1 = 1, \\ g_n &= g_{n-1} + 2g_{n-2} + (-1)^n \ \ dla \ \ n \geq 2. \end{cases}$$ □

Definicja 6.3.7. *Równanie postaci*

$$r_n + pr_{n-1} + qr_{n-2} = 0, n \geq n_0,$$

przy dowolnych warunkach brzegowych postaci $r_{n_o-1} = \alpha, r_{n_o-2} = \beta$, *nazywamy rekurencją liniową jednorodną rzędu drugiego.* □

W tym podrozdziale omawialiśmy tylko sytuację wielomianów o współczynnikach rzeczywistych unikając rozszerzeń na liczby zespolone. Rozdział ten zakończymy jednak wynikiem pokazującym jakie rozwiązania równań rekurencyjnych pojawiają się gdy mianownik ma pierwiastki zespolone (z podobną sytuacją mieliśmy do czynienia w Przykładzie 6.3.6).

Uwaga 6.3.13. *Równanie rekurencyjne liniowe jednorodne rzędu pierwszego ma rozwiązania w zależności od zachowania trójmianu kwadratowego*

$$x^2 + px + q = 0. \tag{6.3.3}$$

Jeśli

- *to równanie ma dwa różne pierwiastki rzeczywiste* $\Delta = p^2 - 4q > 0, \alpha_1 = (-p - \sqrt{\Delta})/2, \alpha_2 = (-p + \sqrt{\Delta})/2$, *to rozwiązanie tego równania jest postaci:*

$$r_n = A_1\alpha_1^n + A_2\alpha_2^n, n \geq n_0, \tag{6.3.4}$$

dla pewnych stałych A_1, A_2,
- *to równanie ma jeden pierwiastek rzeczywisty* $\triangle = p^2 - 4q = 0$, *to rozwiązanie tego równania jest postaci:*

$$r_n = (A_1 n + A_2)(\frac{-p}{2})^n, \tag{6.3.5}$$

dla pewnych stałych A_1, A_2,
- *to równanie ma dwa różne pierwiastki zespolone (jest wielomianem pierwszym)* $\alpha_1 = r(\cos(\phi) + i\sin(\phi)), \alpha_2 = r(\cos(\phi) - i\sin(\phi))$, *to rozwiązanie tego równania jest postaci:*

$$r_n = r^n(A_1\cos(n\phi) + A_2\sin(n\phi)), n \geq n_0,$$

dla pewnych stałych A_1, A_2. □

Stałe A_1 *i* A_2 *wyliczamy z warunków brzegowych, stosując odpowiedni ze wzorów (6.3.3)-(6.3.5) do* $r_{n_o-1} = \alpha$ *oraz* $r_{n_o-2} = \beta$, *i rozwiązując układ dwóch równań z dwoma niewiadomymi.* □

Podsumowanie.

- W rozdziale tym poznaliśmy najsilniejszą metodę rozwiązywania rekurencji $r_n = f_n(r_{n-1}, r_{n-2}, ...)$ takich, że funkcje $f_n(x_1, x_2, ...)$ są liniowe względem swoich argumentów - metodę funkcji tworzących.
- Stosunko łatwo, w tej metodzie, wyliczyć funkcję tworzącą $R(x)$, główną trudnością jest rozwinięcie tej funkcji w szereg potęgowy. Dlatego rozwinęliśmy w tym rozdziale teorię wielomianów oraz funkcji wymiernych (funkcje tworzące są takiej postaci) pokazując jak rozbić je na ułamki proste i jak te ułamki proste rozwinąć w szereg potęgowy.

Rozdział 7

Wstęp do analizy algorytmów

7.1. Prawdopodobieństwo dyskretne **120**
7.2. Asymptotyka **125**
7.3. Algorytmy sortowania **129**
7.4. Problem rozmieszczenia rekordów w pamięci **137**

7.1. Prawdopodobieństwo dyskretne

Doświadczeniem losowym nazywamy dowolne fizyczne lub myślowe doświadczenie które może przyjmować mierzalne wyniki. Przestrzenią zdarzeń elementarnych $\Omega = \{\omega_1, \omega_2, ..., \omega_n\}$ jest to zbiór wszystkich możliwych wyników tego doświadczenia a n jest to ilość możliwych wyników. Liczba n może być skończona lub nieskończona ale w tym rozdziale zakładać będziemy, że zbiór elementów Ω zawsze da się ustawić w ciąg, to znaczy Ω jest zbiorem przeliczalnym. Przestrzeń tą nazywamy wtedy dyskretną (stąd prawdopodobieństwo dyskretne).

Zdarzenie losowe, poza przestrzenią losową, charakteryzuje również prawdopodobieństwo. Dla $\Omega = \{\omega_n, n \in \mathcal{N}\}$ jest to zbiór nieujemnych liczb $\mathcal{P} = \{p_n, n \in \mathcal{N}\}$ skojarzonych z Ω i takich, że $\sum_{n=1}^{\infty} p_n = 1$. O zbiorze $\mathcal{P}$ mówimy, że jest to rozkład prawdopodobieństwa, p_n jest traktowane jako prawdopodobieństwo zdarzenia $\{\omega_n\}$.

Przykład 7.1.1. *Niech doświadczenie losowe polega na rzucie kostką do gry. Wówczas mamy możliwych* 6 *wyników doświadczenia:* $\Omega = \{\omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6\}$ *gdzie* ω_i *oznacza, że na kostce wypadło* i *oczek (*$1 \leq i \leq 6$*). W zależności od tego, czy kostka jest symetryczna, czy taka, że na przykład,* 6 *wypada dwa razy częściej niż inny wynik, możemy określić różne rozkłady prawdopodobieństwa. W pierwszej sytuacji kładziemy* $P(\omega_i) = \frac{1}{6}, 1 \leq i \leq 6$, *a w drugim* $P'(\omega_i) = \frac{1}{7}, 1 \leq i \leq 5, P'(\omega_6) = \frac{2}{7}$. *Trudno jest na pierwszy rzut oka rozstrzygnąć, o jaki rozkład prawdopodobieństwa w naszym konkretnym przypadku chodzi. Możemy wybrać prawidłowy rozkład albo analizując bardzo dokładnie kształt i budowę kostki, albo wykonać pewną liczbę doświadczeń (rzutów kostką) i przeanalizować częstości z jakimi będą się pojawiały poszczególne wartości. Jeżeli kostka jest prawie idealnie symetryczna, to chociaż trudno oczekiwać, że rozkład wszystkich prawdopodobieństw będzie taki jak* P *powyżej, to błąd jaki popełnimy wybierając* P *będzie przypuszczalnie niewielki.*

Rozważmy teraz zdarzenie losowe A*, polegające na tym, że wypadła parzysta liczba oczek, tzn.* $A = \{\omega_2, \omega_4, \omega_6\}$. *Wtedy dla kostki o rozkładzie prawdopodobieństwa* P *mamy* $P(A) = P(\omega_2) + P(\omega_4) + P(\omega_6) = \frac{3}{6} = \frac{1}{2}$, *podczas gdy dla rozkładu* P' *mamy* $P'(A) = P'(\omega_2) + P'(\omega_4) + P'(\omega_6) = \frac{4}{7}$. □

Przykład 7.1.2. *Jeżeli rzucamy po kolei dwoma kostkami, obie idealnie symetryczne (o rozkładzie* P *z Przykładu 7.1.1) i interesuje nas zdarzenie losowe (*A*) polegające na tym, że w sumie na obu kostkach wypadło sześć oczek, to analizę można przedstawić następująco:*

$\Omega = \{\omega_{(i,j)}, 1 \leq i, j \leq 6\}$ *przy czym np.* $\omega_{(3,5)}$ *oznacza, że na pierwszej kostce wypadły* 3 *oczka a na drugiej* 5 *(odróżniamy to zdarzenie od* $\omega_{(5,3)}$*).*

Wtedy $A = \{\omega_{(1,5)}, \omega_{(2,4)}, \omega_{(3,3)}, \omega_{(4,2)}, \omega_{(5,1)}\}$ *i* $P(A) = \frac{5}{36}$. *Rozważmy teraz zdarzenia* $\Omega' = \{\omega'_1, ..., \omega'_6\}$ *- wynik rzutu pierwszą kostką i* $\Omega'' = \{\omega''_1, ..., \omega''_6\}$ *- wynik rzutu drugą kostką, wtedy* $\Omega = \Omega' \times \Omega''$ *(iloczyn kartezjański) tak, że* $\omega_{(i,j)} = (\omega'_i, \omega''_j), 1 \leq i, j \leq 6$. *Niech teraz* $X_1 : \Omega' \to \{1, 2, 3, 4, 5, 6\}$ *tak, że* $X_1(\omega'_i) = i, 1 \leq i \leq 6$, *oraz* $X_2 : \Omega'' \to \{1, 2, 3, 4, 5, 6\}$ *tak, że* $X_2(\omega''_i) = i, 1 \leq i \leq 6$. *Wtedy pytamy się o sumę tych funkcji* $S = X_1 + X_2$. *Zauważmy, że z Tablicy 7.1*

Tabela 7.1. Rozkład sumy zmiennych losowych

S	2	3	4	5	6	7	8	9	10	11	12
	(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)
		(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(3,5)	(4,5)	(5,5)	(6,5)	
			(3,1)	(3,2)	(3,3)	(3,4)	(4,4)	(5,4)	(6,4)		
				(4,1)	(4,2)	(4,3)	(5,3)	(6,3)			
					(5,1)	(5,2)	(6,2)				
						(6,1)					
$P \times P$	$\frac{1}{36}$	$\frac{2}{36}$	$\frac{3}{36}$	$\frac{4}{36}$	$\frac{5}{36}$	$\frac{6}{36}$	$\frac{5}{36}$	$\frac{4}{36}$	$\frac{3}{36}$	$\frac{2}{36}$	$\frac{1}{36}$
$P \times P'$	$\frac{1}{42}$	$\frac{2}{42}$	$\frac{3}{42}$	$\frac{4}{42}$	$\frac{5}{42}$	$\frac{7}{42}$	$\frac{6}{42}$	$\frac{5}{42}$	$\frac{4}{42}$	$\frac{3}{42}$	$\frac{2}{42}$
$P' \times P$	$\frac{1}{42}$	$\frac{2}{42}$	$\frac{3}{42}$	$\frac{4}{42}$	$\frac{5}{42}$	$\frac{7}{42}$	$\frac{6}{42}$	$\frac{5}{42}$	$\frac{4}{42}$	$\frac{3}{42}$	$\frac{2}{42}$
$P' \times P'$	$\frac{1}{49}$	$\frac{2}{49}$	$\frac{3}{49}$	$\frac{4}{49}$	$\frac{5}{49}$	$\frac{8}{49}$	$\frac{7}{49}$	$\frac{6}{42}$	$\frac{5}{42}$	$\frac{4}{49}$	$\frac{4}{49}$

możemy odczytać $P[S = 6] = \frac{5}{36}$. □

Na przestrzeni zdarzeń elementarnych Ω możemy zdefiniować różne funkcje X, Y, Z będące naszym subiektywnym (w odróżnieniu od rozkładu prawdopodobieństwa) wartościowaniem zdarzeń elementarnych, dla jednej osoby wartość zdarzenia ω_1 może wynosić x_1 a dla drugiej y_1. Funkcje te nazywamy zmiennymi losowymi. Funkcje X_1, X_2, S z Przykładu 7.1.2, są to też zmienne losowe.

Funkcja X przyjmująca tylko wartości 0 i 1 nazywana jest indykatorem zdarzenia A gdzie $A = \cup_{\omega \in \Omega: X(\omega)=1} \omega$ i oznaczana jest przez $I[A]$.

Gdy chcemy ocenić zachowanie się jakiejś zmiennej często pytamy o jej wartość przeciętną. Nie jest to jednak wielkość jednoznaczna, możemy obliczać

medianę zmiennej losowej X - jest to każda liczba x taka, że

$$P[X \leq x] \geq \frac{1}{2}, \text{ oraz } P[X \geq x] \geq \frac{1}{2};$$

dominantę zmiennej losowej X - jest to każda liczba rzeczywista x taka, że

$$\forall_{y \in \Re} P[X = x] \geq P[X = y];$$

wartość oczekiwaną zmiennej losowej X - jest to suma

$$EX = \sum_{k \in \mathcal{N}} X(\omega_k) p_k,$$

o ile oczywiście ta suma istnieje.

średnią geometryczną lub harmoniczną zmiennej X to

$$G = \prod_{k \in \mathcal{N}} X(\omega_k)^{1/p_k}$$

i

$$H = \frac{1}{\sum_{k \in \mathcal{N}} 1/p_k X(\omega_k)},$$

odpowiednio. Średnich tych bardzo żadko się używa, jednak są sytuacje, na przykład gdy X jest jakimś ilorazem, indeksem np. BMI, wtedy średnia geometryczna jest najsensowniejszym opisem zmiennej losowej X.

Największe znaczenie spośród wymienionych współczynników ma wartość oczekiwana, chociaż najciekawsze są własności matematyczne mediany. Wartość oczekiwana mierzy przeciętny poziom funkcjii X. Jeżeli zmienną losową X traktować jak grę z jakimś realnym lub abstrakcyjnym przeciwnikiem, a jej wartości to "nasza" wygrana, to wartość oczekiwana jest przeciętną wygraną w jednej grze. Mówi nam ona, czy gra jest dla nas opłacalna (wartość oczekiwana dodatnia), czy sprawiedliwa (wartość oczekiwana 0), czy też nieopłacalna (wartość oczekiwana ujemna). Jednak poza globalną oceną gry chcielibyśmy mieć ocenę ryzyka przystąpienia do pojedynczej gry. Również tutaj mamy bardzo dużo różnych współczynników (np. odchylenie ćwiartkowe, odchylenie od średniej) my jednak skoncentrujemy się tutaj tylko na jednaj - wariancji i jej znormalizowanej postaci - odchyleniu standardowym.

Wariancją zmiennej losowej X nazywamy wartość

$$Var(X) = \sum_{k \in \mathcal{N}} (X(\omega_k) - EX)^2 p_k = \sum_{k \in \mathcal{N}} X(\omega_k)^2 p_k - (EX)^2 = E(X^2) - (EX)^2,$$

o ile również istnieje. Odchylenie standardowe to $\delta(X) = \sqrt{Var(X)}$.

Jeśli $\Omega = \{\omega_1, ..., \omega_n\}$ oraz $p_i = \frac{1}{n}, 1 \leq i \leq n$, a zmienna losowa $X(\omega_i) = i, 1 \leq i \leq n$, to mówimy że zmienną losową X ma rozkład jednostajny na $\{1, 2, ..., n\}$ (piszemy $U(1, n)$) a jeśli $X(\omega_i) = i - 1, 1 \leq i \leq n$, to X ma rozkład jednostajny na $\{0, 1, 2, ..., n-1\}(U(0, n-1))$. Jak łatwo sprawdzić

$$\begin{aligned} EU(1,n) &= \sum_{i=1}^{n} i \frac{1}{n} = \frac{n+1}{2}, \\ EU(0,n-1) &= \sum_{i=0}^{n-1} i \frac{1}{n} = \frac{n-1}{2}. \end{aligned}$$

Przykład 7.1.3. *W jednym z N pączków ukryta jest złota obrączka. Ile powinniśmy zjeść pączków aby tę obrączkę znaleźć? Inaczej mówiąc na której pozycji może się znaleźć pączek ze złotą obrączką? Różnych ustawień zbioru N elementowego jest $N!$ a jeśli pączek jest na i-tej pozycji to różnych ustawień jest $(N-1)!$ (permutacje wszystkich z wyjątkiem tego z obrączką) a więc jeśli zmienna losowa określa numer na której pozycji jest poszukiwany pączek to $P[X=i]=\frac{(N-1)!}{N!}=\frac{1}{N}, 1\leq i\leq N$, a więc*

$$EX=\sum_{i=1}^{N} i\frac{1}{N}=\frac{N(N+1)}{2N}=\frac{N+1}{2}.$$

Łatwo widać w tym przykładzie, że X ma rozkład jednostajny na $\{1,2,3,...,N\}$. □

Mówimy, że dwie rodziny zmiennych losowych $\{X_i, 1\leq i\leq n\}$ oraz $\{Y_i, 1\leq i\leq n\}$ są niezależne , jeżeli dla dowolnych rodzin podzbiorów $\{A_i, 1\leq i\leq n\}$ oraz $\{B_i, 1\leq i\leq n\}$ zbioru Ω mamy

$$P[\cap_{i=1}^{n} X_i\in A_i\cap Y_i\in B_i]=P[\cap_{i=1}^{n} X_i\in A_i]P[\cap_{i=1}^{n} Y_i\in B_i].$$

Twierdzenie 7.1.1. *Dla dowolnych dwóch zmiennych losowych:*

$$\begin{aligned} E(X+Y) &= EX+EY,\\ Var(X+Y) &= Var(X)+Var(Y)+2(E(XY)-(EX)(EY)),\end{aligned}$$

a jeśli X i Y są niezależne, to

$$\begin{aligned} E(XY) &= EX\cdot EY,\\ Var(X+Y) &= Var(X)+Var(Y),\\ Var(XY) &= (EX^2)Var(Y)+(EY)^2Var(X).\end{aligned}$$

Ponadto

$$\begin{aligned} EI[A] &= P(A),\\ Var(I[A]) &= P(A)(1-P(A)).\end{aligned}$$ □

Wniosek 7.1.1. *Jeżeli $\{X_i, i\geq 1\}$ jest ciągiem zmiennych losowych a $V=\sum_{i=1}^{n} X_i$ to $EV=\sum_{i=1}^{n} EX_i$.* □

Pozostaje pytanie, jak analizować sytuację EXY, gdy zmienne losowe X i Y są zależne? Jak zmieni się rozkład prawdopodobieństwa zmiennej losowej X gdy o niej coś więcej wiemy?

Przykład 7.1.4. *Kolega rzucił dwoma symetrycznymi kostkami i oświadczył, że suma oczek wyniosła 5. My tego nie widzieliśmy, ale nie mamy*

podstaw koledze nie wierzyć, jakie jest prawdopodobieństwo, że na pierwszej kostce wypadnie i - oczek (X_1), czy dalej $P[X_1 = i] = \frac{1}{6}, 1 \leq i \leq 6$*? Z Tabeli 7.1 widzimy, że teraz nasza przestrzeń zmieniła się, jeżeli* $S = 5$ *to możliwymi wynikami są* $\Omega_{S=5} = \{\omega_{(1,4)}, \omega_{(2,3)}, \omega_{(3,2)}, \omega_{(4,1)}\}$ *i widzimy, że*

$$P[X_1 = i|S = 5] = \begin{cases} \frac{1}{4}, & dla\ i = 1, 2, 3, 4, \\ 0, & dla\ i = 5, 6. \end{cases}$$ □

Ogólnie, jeśli znane jest że zaszło jakieś zdarzenie B (z $P(B) > 0$) wtedy przestrzeń probabilistyczna zmienia nam się tak, że

$$P(A|B) = P_B(A) = \frac{P[A \cap B]}{P(B)},$$

wartość tę nazywamy prawdopodobieństwem warunkowym zajścia zdarzenie A pod warunkiem, że wiemy, że zaszło zdarzenie B.

Twierdzenie 7.1.2. Wzór na prawdopodobieństwo całkowite *Jeżeli* $\{H_i, 1 \leq i \leq n\}$ *jest takim układem zdarzeń losowych, że*

$$\bigcup_{i=1}^{n} H_i = \Omega, \quad H_i \cap H_j = \emptyset, \ dla\ 1 \leq i \neq j \leq n,$$

wtedy dla dowolnego zdarzenia A zachodzi

$$P(A) = \sum_{i=1}^{n} P[A|H_i]P[H_i].$$

Dowód. Ponieważ $A = \cup_{i=1}^{n}(A \cap H_i)$, więc

$$\begin{aligned} P(A) &= \sum_{i=1}^{n} P[A \cap H_i] \\ &= \sum_{i=1}^{n} P[A|H_i]P[H_i]. \end{aligned}$$ □

Przykład 7.1.5. *W k szufladach znajduje się odpowiednio* $m_1, m_2, ..., m_k$ *kapeluszy tak, że w każdej szufladzie są kapelusze określonego koloru. Chcemy znaleźć jakiś jeden z tych* $N = m_1 + m_2 + ... + m_k$ *kapeluszy. Po kolorze wiemy w jakiej jest szufladzie a z szuflady wybieramy kapelusze po kolei losowo, bez zwracania, dopóty dopóki nie znajdziemy szukanego. Ile kapeluszy przejrzymy? Niech* $H_i, 1 \leq i \leq k$, *oznacza zdarzenie, że szukany kapelusz jest w i-tej szufladzie. Oczywiście* $H_i \cap H_j = \emptyset, i \neq j$, $i\ \Omega = \cup_{i=1}^{n} H_i$. *Ponadto* $P[H_i] = \frac{m_i}{N}, 1 \leq i \leq k$. *Z kolei zdarzenie* $P[X = j|H_i] = \frac{1}{m_i}, 1 \leq j \leq$

m_i, gdzie X oznacza za którym razem z odpowiedniej szuflady wyciągnięto szukany kapelusz. Stąd

$$P[X=j]=\sum_{i:m_i\geq j}\frac{1}{N},$$

i

$$\begin{aligned} EX &= \sum_{j=1}^{\max\{m_i,1\leq i\leq k\}}\frac{j\overline{\overline{\{m_i : m_i\geq j, 1\leq i\leq k\}}}}{N} \\ &= \sum_{i=1}^{k}\sum_{j=1}^{m_i}\frac{j}{N}=\sum_{i=1}^{k}\frac{m_i(m_i+1)}{2N}=\sum_{i=1}^{k}E(X|H_i)P[H_i]. \end{aligned}$$ □

W Przykładzie 7.1.4 otrzymaliśmy wzór

$$EX=\sum_{i=1}^{k}E(X|H_i)P[H_i],$$

gdzie $E(X|A)=\sum_{k=1}^{n}x_kP[X=x_k|A]$.

Twierdzenie 7.1.3. *Jeśli zmienna losowa N może przyjmować wartości ze zbioru $\{1,2,3,...,n\}$ a $\{X_i, 1\leq i\leq n\}$ są zmiennymi losowymi niezależnymi o tym samym rozkładzie i niezależnymi od N, to*

$$E\sum_{i=1}^{N}X_i=\sum_{k=1}^{n}E\sum_{i=1}^{k}X_iI[N=k]=\sum_{k=1}^{n}\sum_{i=1}^{k}EX_1P[N=k]=ENEX_1.$$ □

7.2. Asymptotyka

Rozważając różne nieskończone ciągi możemy wprowadzić pewien porządek między ciągami, uwzględniający tylko graniczne zachowanie ciągów:

$$f(n)\prec g(n)\Longleftrightarrow\lim_{n\to\infty}\frac{f(n)}{g(n)}=0.$$

Relacja $\prec$ jest przeciwzwrotna, przeciwsymetryczna i przechodnia. Definicja tej relacji jest poprawna, gdy $g(n)$ jest odgraniczone od 0 dla dostatecznie dużych n. Można ustawić wiele ciągów w szereg wg. tej relacji:

$$1\prec\log\log(n)\prec\log(n)\prec n^{\epsilon}\prec n^{c}\prec n^{\log(n)}\prec c^{n}\prec n^{n}\prec c^{c^{n}}$$

gdzie $0 < \epsilon < 1 < c$. Dla ciągów zbieżnych do 0 też możemy ustalić pewną hierarchię asymptotycznego zachowania:

$$f(n) \prec g(n) \Longleftrightarrow \frac{1}{g(n)} \prec \frac{1}{f(n)}.$$

Co prowadzi do podobnej hierarchii ciągów

$$\frac{1}{c^{c^n}} \prec \frac{1}{n^n} \prec \frac{1}{c^n} \prec \frac{1}{n^{\log(n)}} \prec \frac{1}{n^c} \prec \frac{1}{n^\epsilon} \prec \frac{1}{\log(n)} \prec \frac{1}{\log\log(n)} \prec 1.$$

Oczywiście zawsze gdy $f(n) \to 0$ a $g(n) \not\to 0$ to $f(n) \prec g(n)$. Mówimy, że $f(n)$ i $g(n)$ mają ten sam współczynnik przyrostu $f(n) \asymp g(n)$, jeżeli istnieje taka stała C, że $|f(n)| \leq C|g(n)|$ oraz $|g(n)| \leq C|f(n)|$. Silniejszy od tego samego współczynnika przyrostu związek - mówimy, że $f(n)$ jest asymptotyczna do $g(n)$ ($f(n) \sim g(n)$) jeśli $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$. Symbolu tego użyliśmy w Twierdzeniu 4.2.5 dla opisania asymptotycznego zachowania gęstości i wartości liczb pierwszych w zbiorze liczb naturalnych. Z Przykładu 6.2.1 dla liczb Fibonacciego F_n mamy $F_n \asymp (\frac{1+\sqrt{5}}{2})^n$. Z kolei, z podrozdziału 2.4 mamy $H_n \prec \ln(n)$.

W roku 1894 niemiecki matematyk Paul Bachmann wprowadził konwencję notacyjną zwaną "duże O".

$$f(n) = O(g(n)) \Longleftrightarrow \exists_C \forall_{n\in\mathcal{N}} |f(n)| \leq C|g(n)|.$$

Na przykład
w Przykładzie 1.2.2 suma ciągu arytmetycznego spełnia $s_n = O(n^2)$.
w Przykładzie 1.2.4 ilość ruchów w problemie wieży z Hanoi $h_n = O(2^n)$.
w Przykładzie 1.2.5 ilość obszarów w problemie pizzy $l_n = O(n^2)$.
w Przykładzie 1.2.6 pozycja w problemie Flawiusza $J_s(n) = O(n)$, ale ten wynik akurat jest kompletnie bezwartościowy.
w Przykładzie 1.2.3 w problemie cyfr $a_n = O(10^n)$.
w Przykładzie 2.2.1 ciągu τ_n spełnia $\tau_n = O(n\ln(n))$.
w Rozdziale 2.4 dla ciągu H_n mamy

$$\begin{aligned} H_n &= O(\ln(n)), \\ H_n - \ln(n) &= O(1), \\ H_n - \ln(n) - \gamma &= O(\frac{1}{n}), \\ H_n - \ln(n) - \gamma - \frac{1}{2n} &= O(\frac{1}{n^2}) \end{aligned}$$

i tak dalej.
w Przykładzie 3.1.1 jest $\sum_{0\leq k<n} \lfloor \frac{k}{2} \rfloor = O(n^2)$.

Pojawiają się pewne wątpliwości w chwili gdy używamy symbolu O po obu stronach równości, na przykład, co oznacza zapis

$$f(n) + O(g(n)) = O(h(n)).?$$

Formalnie biorąc symbol $O(g(n))$ oznacza zbiór takich wszystkich takich ciągów, które podzielone przez $g(n)$ są w granicy ograniczone, a równość $f(n) = O(g(n))$ nie oznacza, że zbiór $O(g(n))$ jest jednoelementowy i zawiera tylko $f(n)$, lecz że $f(n) \in O(g(n))$. Wtedy zapis $f(n) + O(g(n)) = O(h(n))$ możemy interpretować tak, że jeśli $O(g(n)) = \{g_x(n) : x \in J\}$ to $\{f(n) + g_x(n) : x \in J\} \subset O(h(n))$. Przy tym zastrzeżeniu możemy używać symbolu O po obu stronach równości.

Twierdzenie 7.2.1. *Dla dowolnych ciągów $f(n), g(n)$, zachodzi*

$$\begin{aligned}
O(f(n)) + O(g(n)) &= O(f(n) + g(n)),\\
f(n) &= O(f(n)),\\
cO(f(n) &= O(f(n)),\\
O(O(f(n))) &= O(f(n)),\\
O(f(n))O(g(n)) &= O(f(n)g(n)),\\
O(f(n)g(n)) &= f(n)O(g(n)),\\
\ln(1 + O(f(n))) &= O(f(n)),\\
e^{O(f(n))} &= 1 + O(f(n)).
\end{aligned}$$

□

Edmund Landau zaproponował następującą notację "małego o":

$$\begin{aligned}
f(n) = o(g(n)) &\iff \forall_{\epsilon>0}\exists_{n_o\in\mathcal{N}}\forall_{n\geq n_o}|f(n)| \leq \epsilon|g(n)|\\
&\iff f(n) \prec g(n).
\end{aligned}$$

Symbol $o(f(n))$ jest również bardzo popularny, my jednak do końca tego rozdziału skupimy się na symbolu $O(f(n))$.

Jak w praktyce szacować asymptotyczne zachowanie ciągu? Bardzo często rozważany ciąg jest ciągiem sum częściowych, tzn. $f(n) = \sum_{k=1}^{n} a_k$, jak wtedy postępować? Pomóc nam mogą następujące wyniki:

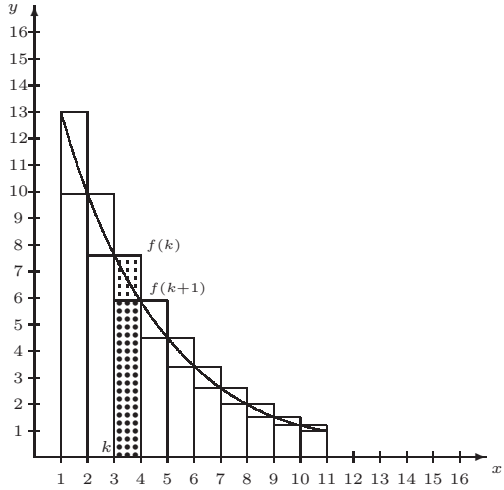
Twierdzenie 7.2.2. Cauchy'ego - Maclaurina. *Jeżeli rzeczywista nieujemna funkcja $f(x)$ jest malejąca (rosnąca) w przedziale $[a, b]$ dla $a, b \in \mathcal{Z}$ to*

$$\sum_{k=a+1}^{b} f(k) \leq \int_a^b f(x) \leq \sum_{k=a}^{b-1} f(k),$$

($\sum_{k=a}^{b-1} f(k) \leq \int_a^b f(x) \leq \sum_{k=a+1}^{b} f(k)$.)

Twierdzenie to wynika z faktu, że interpretacją geometryczną całki jest pole powierzchni poniżej wykresu tej funkcji. Z Rysunku 7.1 łatwo sprawdzić, że pole $\int_k^{k+1} f(x)\,dx$ ograniczają pola dwóch prostokątów $f(k)\cdot 1$ od góry i $f(k+1)\cdot 1$ od dołu.

Rysunek 7.1. Twierdzenie Cauchy'ego - Maclaurina.



Wniosek 7.2.1. *Ponieważ*

$$\begin{aligned}
\int x^a\,dx &= \begin{cases} \frac{1}{a+1}x^{a+1}, & dla\ a \neq -1, \\ \ln(x), & dla\ a = -1, \end{cases} \\
\int \ln(x)\,dx &= (x-1)\ln(x), \\
\int \frac{1}{x(\ln(x))^m}\,dx &= \frac{-1}{(m-1)(\ln(x))^{m-1}},
\end{aligned}$$

więc

$$\begin{aligned}
\sum_{k=1}^{n} k^a &= \begin{cases} O(n^{a+1}), & dla\ a > -1, \\ O(\ln(n)), & dla\ a = -1, \\ O(1), & dla\ a < -1, \end{cases} \\
\sum_{k=1}^{n} \ln(k) &= O(n\ln(n)), \\
\sum_{k=1}^{n} \frac{1}{k(\ln(k))^m} &= O(\frac{1}{(\ln(n))^{m-1}}).
\end{aligned}$$

□

Przykład 7.2.1. *Rozważmy sumę z Przykładu 3.3.1. Ponieważ*

$$O(n\ln(n)) = \frac{1}{\ln(2)}\sum_{k=1}^{n}\ln(k) < \sum_{k=1}^{n}\lfloor\log_2(k)\rfloor < \frac{1}{\ln(2)}\sum_{k=1}^{n}\ln(k)+n = O(n\ln(n))+n,$$

a więc

$$\sum_{k=1}^{n} \lfloor \log_2(k) \rfloor = O(n \ln(n)). \qquad \square$$

W przypadku sum zawierających symbol silni można skorzystać z następującego opisu asymptotycznego zachowania silni:

Twierdzenie 7.2.3. Wzór Stirlinga *Zachodzą*

$$n! \sim (\frac{n}{e})^n \sqrt{2\pi n},$$

oraz

$$e^{\frac{k(k-1)}{2n}} \leq \frac{n^k}{n^{\underline{k}}} \leq e^{\frac{k(k-1)}{2(n-k+1)}}. \qquad \square$$

7.3. Algorytmy sortowania

Przedstawimy tutaj przykład analizy algorytmów porządkowania ciągu.

7.3.1. Sortowanie przez wstawianie (insertion sort)

Sama idea algorytmu sortowania przez wstawianie polega na przeniesieniu i -tego elementu i wstawienie w odpowiednie miejsce ciągu $a_1, a_2, ..., a_{i-1}$.

Tabela 7.2. Działanie algorytmu proste wstawianie.

Początkowy ciąg	32	45	15	27	77	21	11	73
$i = 2$	32	45	15	27	77	21	11	73
$i = 3$	15	32	45	27	77	21	11	73
$i = 4$	15	27	32	45	77	21	11	73
$i = 5$	15	27	32	45	77	21	11	73
$i = 6$	15	21	27	32	45	77	11	73
$i = 7$	11	15	21	27	32	45	77	73
$i = 8$	11	15	21	27	32	45	73	77

Metoda ta jest powszechnie stosowana przez grających w karty (do szybkiego układania kart). Algorytm można opisać następująco:

`Dla` $i = 2, 3, 4, ...n$

`Weź element` a_i `i wstaw go w odpowiednie miejsce do ciągu` $a_1, a_2, ..., a_{i-1}$

a szczegółowo:

Algorytm 7.1. Algorytm **Proste wstawianie**

```
procedure prostewstawianie();
    i,j: indeks;
    x : obiekt;
    for i:=2 to n do
        {
          x:=a[i]; a[0]:=x; j:=i-1;
          while x.klucz<a[j].klucz do
                  { a[j+1]:=a[j]; j:=j-1; }
          a[j+1]:=x;
         }
```

Analiza algorytmu 7.1. Załóżmy dla uproszczenia że ciąg $\{a_j, 1 \leq j \leq n\}$ jest ciągiem o parami różnych wyrazach. Spośród każdego wyboru i elemementów takiego ciągu (jest tych wyborów $C_i^n = \binom{n}{i}$), oznaczmy jeden konkretny wybór przez $x_1 < x_2 < ... < x_i$. Spośród ustawień tych elementów w ciąg jest dokładnie jedno ustawienie takie, że elementy od 1 do $i-1$ są rosnące a i ty element jest większy od $k-1$ i mniejszy od k -tego ($1 \leq k \leq i$). Jest to ustawienie $\{x_1, x_2, ..., x_{k-1}, x_{k+1}, x_{k+2}, ..., x_i, x_k\}$. Stąd prawdopodobieństwo, że i ty element będzie wstawiony między $k-1$ a k-tym wynosi $\frac{1}{i}$. Wtedy wykonujemy w pętli

```
while x.klucz<a[j].klucz do
```

$i-k$ porównań, a więc najmniejsza liczba porównań w i-tym obrocie pętli zewnętrznej Po_{min} wynosi 1, średnia Po_{sr} jest równa $\sum_{k=1}^{i-1}(i-k)\frac{1}{i} = \frac{(i-1)i}{2i} = \frac{i-1}{2}$ natomiast w najgorszym przypadku $Po_{max} = i-1$. Sumując po i otrzymujemy

$$\begin{array}{lllllll} Po_{min} & = & \sum_{i=2}^{n} 1 & = & n-1 & = & O(n), \\ Po_{sr} & = & \sum_{i=2}^{n} \frac{i-1}{2} & = & \frac{(n-1)n}{4} & = & O(n^2), \\ Po_{max} & = & \sum_{i=2}^{n}(i-1) & = & \frac{(n-1)n}{2} & = & O(n^2), \end{array}$$

a uwzględniając, że liczba przestawień $P_w = Po + 2$ otrzymujemy

$$\begin{array}{lllllllll} Po_{min} & = & n-1 & = & O(n), & Pw_{min} & = & n+1 & = & O(n), \\ Po_{sr} & = & \frac{n^2-n}{4} & = & O(n^2), & Pw_{sr} & = & \frac{n^2-n+8}{4} & = & O(n^2), \\ Po_{max} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{max} & = & \frac{n^n-n+4}{2} & = & O(n^2). \end{array}$$ □

Algorytm 7.1 można poprawić, zauważając, że w chwili wstawiania i-tego elementu do ciągu wynikowego $a_1 \leq a_2 \leq ... \leq a_{i-1}$ elementy te są już

uporządkowane, a wtedy najszybszą metodą ustalenie miejsca wstawianego elementu uzyskuje się przez porównanie z elementem środkowym ciągu $\{a_1, ..., a_{i-1}\}$. W zależności od wyniku porównania porównujemy dalej z elementem środkowym ciągu $\{a_1, ..., a_{\lfloor \frac{i-1}{2} \rfloor}\}$ lub $\{a_{\lceil \frac{i-1}{2} \rceil}, ..., a_{i-1}\}$ kontunuując ten proces dalej.

Algorytm 7.2. Algorytm **Wstawianie połówkowe**

```
procedure wstawianiepołówkowe();
  i,j,l,p,m: indeks;
  x : obiekt;
  for i:=2 to n do
     {
     x:=a[i]; l:=1; p:=i-1;
     while l<=p do
          {  m:=(l+p) div 2;
             if x.klucz<a[m].klucz then p:=m-1 else l:=m+1;
          }
     for j:=i-1 downto l do a[j+1]:=a[j];
     a[l]:=x;
     }
```

Analiza algorytmu 7.2. Jak łatwo widać połowienie przedziału złożonego z i-kluczy wykonuje się $\lceil \log_2(i) \rceil$ razy, a więc liczba porównań w tym algorytmie wynosi

$$Po_{sr} = \sum_{k=1}^{n} \lceil \log_2(k) \rceil.$$

Suma ta pojawiła się już w naszych rozważaniach w Przykładzie 3.3.1. Aproksymując tę sumę całką $\int_1^n log(x)\,dx = x(log(x-c)|_1^n = O(n\log(n))$. Niestety inne parametry pozostają bez zmian.

$$\begin{array}{lllllllll} Po_{min} & = & n-1 & = & O(n), & Pw_{min} & = & n+1 & = & O(n), \\ Po_{sr} & = & \sum_{k=1}^{n} \lceil \log_2(k) \rceil & = & O(n\ln(n)), & Pw_{sr} & = & \frac{n^2-n+8}{4} & = & O(n^2), \\ Po_{max} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{max} & = & \frac{n^n-n+4}{2} & = & O(n^2). \end{array}$$

□

7.3.2. Algorytmy sortowania przez wybieranie (selection sort)

Algorytm ten polega na znajdywaniu najmniejszego klucza i wstawieniu go w odpowiednie miejsce. Algorytm można opisać następująco:

`Dla` $i = 1, 3, 4, ...n$

`Znajdź najmniejszy element ciągu` $\{a_i, a_{i+1}, ..., a_n\}$

`i wymień go z elementem` a_i

na przykładzie

Tabela 7.3. Działanie algorytmu - proste wybierania

Początkowy ciąg	32	45	15	27	77	21	11	73
	11	45	15	27	77	21	32	73
	11	15	45	27	77	21	32	73
	11	15	21	27	77	45	32	73
	11	15	21	27	77	45	32	73
	11	15	21	27	32	45	77	73
	11	15	21	27	32	45	73	77

a sam algorytm można opisać:

Algorytm 7.3. Algorytm **proste wybieranie**

```
procedure prostewybieranie();
  i,j,k: indeks;
  x : obiekt;
  for i:=1 to n-1 do
      {
        k:=i; x:=a[i];
        for j:=i+1 to n do
            if a[i].klucz<x.klucz then
                { k:=j; x:=a[j]; }
        a[k]:=a[i]; a[i]:=x;
       }
```

Analiza algorytmu 7.3. Jak łatwo widać liczba porównań wogóle nie zależy od początkowego ustawienia ciągu i jest równa $Po = \frac{1}{2}(n^2 - n)$, natomiast najmniejsza liczba wymian (podstawień) wynosi $3(n-1)$, gdy ani razu nie nie będzie spełniony warunek `a[i].klucz<x.klucz` a największa $\lfloor \frac{n^2}{4} \rfloor + 3(n-1)$ gdy ten warunek będzie spełniony zawsze. Wartość średnia ilości wymian jest trudna do ocenienia. Niech X_j oznacza ilość ustawień ciągu n elementów $\{1, 2, 3, ..., n\}$ takich, że k_j jest mniejszy od wszystkich poprzedzających je liczb $k_1, ..., k_{j-1}$. Knuth [11] wykazał, że $EX_j = H_j - 1$. Wtedy $Pw_{sr} = \sum_{i=1}^{n}(H_i - 1)$, a z wyników podrozdziału 2.4 (Uwaga 2.4.4)

uzyskujemy $Pw_{sr} \approx \sum_{i=1}^{n}(\ln(i)+\gamma+1) = O(n\ln(n))$. W końcu

$$\begin{array}{lllllllllll} Po_{min} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{min} & = & 3n-3 & = & O(n), \\ Po_{sr} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{sr} & = & \sum_{i=1}^{n}(H_i-1) & = & O(n\ln(n)), \\ Po_{max} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{max} & = & \lfloor\frac{n^2}{4}\rfloor+3n-3 & = & O(n^2). \quad \square \end{array}$$

7.3.3. Sortowanie przez zamianę (exchange sort)

Najbardziej znaną metodą w tej grupie metod jest tak zwane sortowanie bąbelkowe. Polega ona na ciągłym porównywaniu i zamianie (jeśli pierwszy wyraz jest większy od drugiego) dwóch kolejnych wyrazów ciągu. Opiszemy jego działanie na przykładzie:

Tablica 7.4: Sortowanie bąbelkowe.

$i=1$	32−45	15	27	77	21	11	73	
	32	45↔15	27	77	21	11	73	
	32	15	45↔27	77	21	11	73	
	32	15	27	45−77	21	11	73	
	32	15	27	45	77↔21	11	73	
	32	15	27	45	21	77↔11	73	
	32	15	27	45	21	11	77↔73	
$i=2$	32↔15	27	45	21	11	73	77	
	15	32↔27	45	21	11	73	77	
	15	27	32−45	21	11	73	77	
	15	27	32	45↔21	11	73	77	
	15	27	32	21	45↔11	73	77	
	15	27	32	21	11	45−73	77	
	15	27	32	21	11	45	73−77	
$i=3$	15−27	32	21	11	45	73	77	
	15	27−32	21	11	45	73	77	
	15	27	32↔21	11	45	73	77	
	15	27	21	32↔11	45	73	77	
	15	27	21	11	32−45	73	77	
	15	27	21	11	32	45−73	77	
	15	27	21	11	32	45	73−77	
$i=4$	15−27	21	11	32	45	73	77	
	15	27↔21	11	32	45	73	77	
	15	21	27↔11	32	45	73	77	
	15	21	11	27−32	45	73	77	
	15	21	11	27	32−45	73	77	
	15	21	11	27	32	45−73	77	
	15	21	11	27	32	45	73−77	

$i=5$	15−21	11	27	32	45	73	77	
	15	21↔11	27	32	45	73	77	
	15	11	21↔17	32	45	73	77	
	15	11	21	27−32	45	73	77	
	15	11	21	27	32−45	73	77	
	15	11	21	27	32	45−73	77	
	15	11	21	27	32	45	73−77	
$i=6$	15↔11	21	27	32	45	73	77	
	11	15−21	27	32	45	73	77	
	11	15	21−17	32	45	73	77	
	11	15	21	27−32	45	73	77	
	11	15	21	27	32−45	73	77	
	11	15	21	27	32	45−73	77	
	11	15	21	27	32	45	73−77	
$i=7$	11−15	21	27	32	45	73	77	
	11	15−21	27	32	45	73	77	
	11	15	21−17	32	45	73	77	
	11	15	21	27−32	45	73	77	
	11	15	21	27	32−45	73	77	
	11	15	21	27	32	45−73	77	
	11	15	21	27	32	45	73−77	
$i=8$	11−15	21	27	32	45	73	77	
	11	15−21	27	32	45	73	77	
	11	15	21−17	32	45	73	77	
	11	15	21	27−32	45	73	77	
	11	15	21	27	32−45	73	77	
	11	15	21	27	32	45−73	77	
	11	15	21	27	32	45	73−77	

Algorytm 7.4. Algorytm **Sortowanie bąbelkowe**

```
procedure sortowaniebabelkowe();
   i,j: indeks;
   x : obiekt;
   for i:=2 to n do
      {
        for j:=n downto i do
            if a[j-1].klucz>a[j].klucz then
                  { x:=a[j-1];a[j-1]:=a[j]; a[j]:=x; }
      }
```

Analiza algorytmu sortowania bąbelkowego Liczba porównań w tym algorytmie zawsze wynosi $Po = \frac{n^2-n}{2}$ natomiast najmniejsze liczba przesunięć obiektów wynosi 0 gdy ciąg jest już posortowany, $\sum_{i=2}^{n}\sum_{j=i}^{n} 3 = \sum_{i=2}^{n} 3(n-i+1) = 3n(n-1) - \sum_{i=1}^{n-1} i = \frac{3}{4}(n^2-n)$. Ponieważ każde dwa elementy mają jednakowe prawdopodobieństwo że je wymienimy jak i że pozostawimy bez zmian więc $Pw_{sr} = \frac{1}{2}Pw_{max} = \frac{3}{2}(n^2-n)$.

$$\begin{array}{lllllllll} Po_{min} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{min} & = & 0 & = & o(1), \\ Po_{sr} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{sr} & = & \frac{3}{2}(n^2-n) & = & O(n^2), \\ Po_{max} & = & \frac{n^2-n}{2} & = & O(n^2), & Pw_{max} & = & \frac{3}{4}(n^2-n) & = & O(n^2). \end{array} \quad \square$$

Łatwo widać niedociągnięcia tego algorytmu. Ostatnie dwa obroty zewnętrznej pętli nie powodowały żadnej zmiany ciągu (ciąg był już posortowany), jeżeli najmniejsze wyrazy są na końcu ciągu sortowanie trwa bardzo długo, może więc raz przechodzić w jedną a raz w drugą stronę, po i-tym obrocie pętli zewnętrznej, już i elementów początkowych i końcowych ciągu jest posortowanych. Uwzględniając te uwagi możemy podać inny algorytm:

Algorytm 7.5. Algorytm **Sortowanie mieszane**

```
procedure sortowaniemieszane();
   j,k,l,p: indeks;
   x : obiekt;
   l:=2; p:=n; k:=n;
   repeat
      {
       for j:=p downto l do
            if a[j-1].klucz>a[j].klucz then
                 { x:=a[j-1];a[j-1]:=a[j]; a[j]:=x; k:=j; }
       l:=k+1
       for j:=l to p do
            if a[j-1].klucz>a[j].klucz then
                 { x:=a[j-1];a[j-1]:=a[j]; a[j]:=x; k:=j; }
       p:=k-1;
       } until l>p;
```

Jednak analiza tego algorytmu jest zbyt skomplikowana i wykracza poza ramy tej książki. Wspomnimy, że poprawia się tylko średnia liczba wymian $Pw_{sr} = O(n)$, a wg. Wirtha ([21]) algorytm bąbelkowy nie wyróżnia się niczym poza nazwą.

7.3.4. Sortowanie szybkie (quicksort)

Algorytm ten można opisać następująco:

Krok 1 Porządkowanie elementów a od 1 do n-tego.
Wybierz element środkowy $x = a_{\lfloor \frac{n+1}{2} \rfloor}$ ciągu $\{a_i, 1 \leq i \leq n\}$.
Przeglądaj tablicę od elementu numer 1 do pozycji elementu x włącznie w prawo, aż napotkasz element taki, że $a_i \geq x$.
Przeglądaj tablicę od elementu numer n do pozycji elementu x włącznie w lewo, aż napotkasz element taki, że $a_j \leq x$.
Wtedy wymień elementy a_i oraz a_j i kontunuuj przeglądanie dalej.
W efekcie tej wymiany może się zmieniać pozycja elmentu x w ciągu. Powiedzmy, że w końcu element x jest na pozycji l

Krok 2 Sortowanie podciągów.
Zastosuj całą procedurę sortowania dla elementów od 1 do $l-1$ o ile $l > 2$.
Zastosuj całą procedurę sortowania dla elementów od l+1 do n o ile $n > l + 2$.

a oto przykład jego zastosowania:

Tabela 7.5. Działanie algorytmu - szybkie sortowanie

Początkowy ciąg	32	45	15	|27|	77	21	11	73
	32	21	15	27	77	45	11	73
	(11	|21|	15)	27	(77	|45|	32	73)
	(|15|	11)	21	27	32	45	(|77|	73)
	11	15	21	27	32	45	73	77

Sam algorytm można opisać:

Algorytm 7.6. Algorytm **Quicksort**

```
Wywolanie: sortuj(1,n)

procedure sort(l,p:index);
   i,j; indeks;
   x,w: objekt;
   i:=l; j:=p;
   x=a[(l+p) div 2];
   repeat
      {
        while a[i].klucz<x.klucz do i:=i+1;
        while x.klucz<a[j].klucz do j:=j-1;
        if i<=j then
           {
             w:=a[i]; a[i]:=a[j]; a[j]:=w;
             i:=i+1; j:=j-1;
           }
      }  until i>j;
    if l<j then sortuj(l,j);
    if i<p then sortuj(i,p);
```

Analiza algorytmu quicksort Przeanalizujmy teraz λ_n średnią ilość zamian w tablicy $a[]$ w pierwszym kroku (przy pierwszym przejrzeniu tablicy). Zależy ona istotnie od początkowego układu ciągu $\{a_i, 1 \le i \le n\}$. Załóżmy, że zbiór danych do podziału składa się z ciągu liczb $\{1, 2, 3, ..., n\}$ losowo ustawionych oraz, że wybraną początkową liczbą jest $x \in \{1, 2, 3, ..., n\}$. Wtedy ilość porównań wynosi: ilość elementów ciągu mniejszych od x, czyli $x-1$ razy prawdopodobieństwo zamiany elementu to znaczy prawdopodobieństwo, że element ten jest nie mniejszy niż x, czyli $\frac{n-x+1}{n}$. Zatem ponieważ każda liczba x ma jednakowe prawdopodobieństwo wyboru $\frac{1}{n}$, więc

$$\begin{aligned}
\lambda_n &= \frac{1}{n}\sum_{x=1}^{n}(x-1)\frac{n-x+1}{n} = \frac{1}{n}\sum_{x=1}^{n}(x-1) - \frac{1}{n^2}\sum_{x=1}^{n}x(x-1) + \frac{1}{n^2}\sum_{x=1}^{n}(x-1) \\
&= \frac{1}{n}\sum_{0\le x<n} x^{\underline{1}} - \frac{1}{n^2}\sum_{0\le x<n+1} x^{\underline{2}} + \frac{1}{n^2}\sum_{0\le x<n} x^{\underline{1}} \\
&= \frac{n^{\underline{2}}}{2n} - \frac{(n+1)^{\underline{3}}}{3n^2} + \frac{n^{\underline{2}}}{2n^2} = \frac{n(n-1)}{2n} - \frac{(n-1)(n+1)}{3n} + \frac{n-1}{2n} \\
&= \frac{3n^2-3n-2n^2+2}{6n} + \frac{n-1}{2n} = \frac{n}{6} - \frac{1}{6n}
\end{aligned}$$

Stąd, jeśli k_n jest przeciętną ilością pętli `repeat` w powyższym algorytmie, to $Po_{sr} = (n-1)k_n, Pw_{sr} = (\frac{n}{6} - \frac{1}{6n})k_n$. Niestety, nie potrafimy dokładnie oszacować k_n. Jeżeli będziemy mieć szczęście i za każdym razem wybierać będziemy za x medianę (wartość środkową ciągu) to wtedy liczba obrotów pętli `repeat` wynosić będzie $\log_2(n)$, wtedy $Po_{min} \approx n\log_2(n), Pw_{min} \approx$

$n\log_2(n)$. W najgorszym przypadku ilość podziałów jakie się wykonuje wynosi nie $\log_2(n)$ ale n (segment długości k dzieli się na $k-1$ elementów z jednej strony i 1 element z drugiej strony) stąd $Po_{max} = Pw_{max} = O(n^2)$.

$$\begin{array}{rclrcl} Po_{min} & = & O(n\ln(n)), & Pw_{min} & = & 0(n\ln(n)), \\ Po_{sr} & = & (n-1)k_n, & Pw_{sr} & = & \frac{nk_n}{6}, \\ Po_{max} & = & O(n^2), & Pw_{max} & = & O(n^2). \end{array}$$ □

Uwaga 7.3.1. *Jeżeli zmienimy algorytm quicksort tak, że element x nie jest środkowym elementem ciągu ale jest wybrany losowo z ciągu to można ograniczyć średnią liczbę porównań tak, aby $Po_{sr} < 2\ln(2)Po_{min}, Pw_{sr} < 2\ln(2)Pw_{min}$. Wystarczy w tym celu zmienić w algorytmie 7.6 wiersz 5 z*

```
x=a[(l+p) div 2];
```

na

```
x=a[l+rand(l+p-1)];
```

gdzie funkcja $rand(n)$ losuje liczby $\{0,1,2,3,4,...,n-1\}$ z jednakowym prawdopodobieństwem $\frac{1}{n}$. Dla takiego algorytmu

$$\begin{array}{rclrcl} Po_{min} & = & O(n\ln(n)), & Pw_{min} & = & O(n\ln(n)), \\ Po_{sr} & = & O(n\ln(n)), & Pw_{sr} & = & O(n\ln(n)), \\ Po_{max} & = & O(n^2), & Pw_{max} & = & O(n^2). \end{array}$$ □

7.4. Problem rozmieszczenia rekordów w pamięci

Problem 7.4.1. *W jaki sposób zorganizować w pamięci komputera składowanie i wyszukiwanie informacji? Interesuje nas informacja zorganizowana w postaci dwóch tabel: KLUCZ będący identyfikatorem porcji danych i same DANE. Zakładać będziemy, że wszelkie operacje wykonywalne na rekordach (czyli wyszukiwanie, usuwanie, wstawianie rekordów i modyfikacja rekordu) wykonywane są za pośrednictwem KLUCZA a DANE[KLUCZA] mogą być bardzo nieregularne, ponadto dopuszczamy, że rekordy mogą być zmiennej długości. Dobrym przykładem jest jakiś słownik języka polskiego. Wszystkich możliwych wyrazów (czyli KLUCZY), przyjmując, że w języku polskim nie ma wyrazów dłuższych niż 26 - literowe jest teoretycznie 36^{26}, czyli liczba znacznie przewyższająca możliwości pamięciowe nawet największych komputerów, a DANE jest to opis wyrazu, a więc jak się odmienia, w jakich zwrotach występuje itp.* □

Istnieje kilka rozwiązań tego problemu:

(i) Zbiory sekwencyjne,

(ii) Zbiory indeksowo-sekwencyjne,
(iii) Zbiory indeksowe,
(iv) Listy,
(v) Zbiory uporządkowane wyszukiwane blokowo,
(vi) Zbiory uporządkowane wyszukiwane binarnie,
(vii) Przekształcenia typu $KLUCZ \to ADRES$, na przykład tablice mieszające "hash table".

W każdym zbiorze rekordów chcemy wykonywać cztery podstawowe operacje (poza operacjami zmieniającymi samą strukturę rekordów):

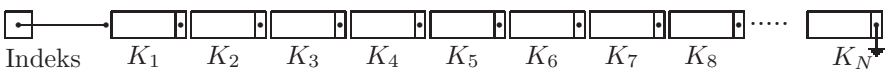
– wyszukiwanie rekordów,
– dodawanie rekordów,
– usuwanie rekordów,
– modyfikacja (zmiana) danych zapisanych w rekordach

Jednak najważniejszą z tych operacji jest operacja wyszukiwania gdyż najczęściej aby dodać, usunąć lub zmodyfikować rekord najpierw trzeba rekord znaleźć! Analiza sposobów rozmieszczenia rekordów polega na ocenie liczby przeszukanych rekordów (najmniejszej, przeciętnej czyli wartości oczekiwanej i największej) w przypadku wyszukiwań zakończonych sukcesem (W_{min}, W_{sr}, W_{max}), liczba przeszukanych rekordów w przypadku braku powodzenia wyszukiwania ($NW_{min}, NW_{sr}, NW_{max}$), liczba rekordów które trzeba przesunąć w nowe miejsce w przypadku dodawania (A_{min}, A_{sr}, A_{max}) oraz usuwania (U_{min}, U_{sr}, U_{max}). Wszystkie te parametry mogą też dotyczyć zbioru indeksów, wtedy odpowiednie nazwy podawać będziemy z indeksem górnym ^{i}. Jeśli wartości minimalne, przeciętne i maksymalne będą równe, piszemy krótko bez indeksu dolnego (np. $U = \frac{1}{N}$). Ilość rekordów zawsze oznaczać będziemy przez N.

7.4.1. Zbiory sekwencyjne i listy

Nieuporządkowane zbiory sekwencyjne.
Rekordy są pamiętane po kolei w postaci pary KLUCZ+DANE[KLUCZA]. Rekord kończy specjalny znak a całą strukturę specjalny końcowy rekord. Zapamiętujemy tylko gdzie się w pamięci cała struktura zaczyna.

Rysunek 7.2. Zbiory sekwencyjne



Ponieważ rekordem wyszukiwanym może być pierwszy rekord w ciągu (sytuacja najlepsza), ostatni (sytuacja najgorsza) a każdy rekord jest wyszukiwany z prawdopodobieństwem $\frac{1}{N}$, stąd

$$W_{min} = 1, \quad W_{sr} = \sum_{i=1}^{N} i\frac{1}{N} = \frac{N+1}{2}, \quad W_{max} = N,$$

a w przypadku niezakończonych powodzeniem wyszukiwań zawsze trzeba przejrzeć wszystkie rekordy, więc $NW = N$. Operacja dodawania nie wymaga zmiany pozycji rekordów, a więc $A = 0$, natomiast znalezienie końca struktury, gdzie powinien być dopisany nowy rekord, wymaga aż przeglądania N rekordów. Operacja usuwania wymaga odpowiednio

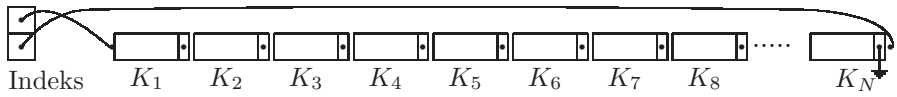
$$U_{min} = 1, \quad U_{sr} = \frac{N+1}{2}, \quad U_{max} = N,$$

przesunięć rekordów. Wszystkie operacje na indeksach to pobranie pojedynczego indeksu.

Nieuporządkowane zbiory sekwencyjne z 2 indeksami.

Aby zmniejszyć ilość wyszukiwań potrzebnych do operacji dodawania, czasem dobrze jest dorzucić wskaźnik do końca struktury, tak jak przedstawiamy to na następującym rysunku:

Rysunek 7.3. Zbiory sekwencyjne z 2 indeksami



Parametry tej organizacji, poza sposobem wyszukiwania przy dodawaniu, nie zmieniają się, ewentualnie można tylko zmniejszyć ilość przesuwanych rekordów przy usuwaniu przesuwając je do przodu lub do tyłu, w zależności od tego czy usuwamy rekord bliższy początku struktury czy końca. Wtedy

$$U_{sr} = \begin{cases} \sum_{i=1}^{N/2} 2i\frac{1}{N}, & \text{jeśli } 2|N, \\ \sum_{i=1}^{\lfloor N/2 \rfloor} 2i\frac{1}{N} + (\lfloor N/2 \rfloor + 1)\frac{1}{N}, & \text{jeśli } 2 \not| N. \end{cases}$$

a więc

$$U_{sr} = \begin{cases} \frac{\lfloor N/2 \rfloor + 1}{2}, & \text{jeśli } 2|N, \\ \frac{(\lfloor N/2 \rfloor + 1)^2}{N}, & \text{jeśli } 2 \not| N. \end{cases}$$

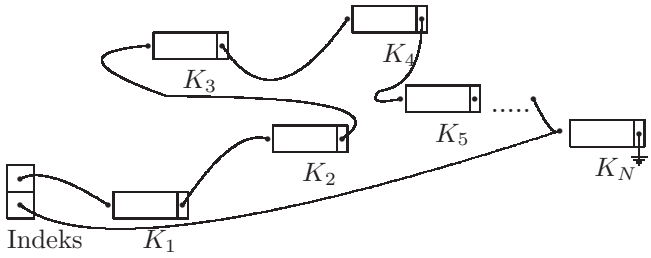
i $U_{max} = \lceil N/2 \rceil$.

Lista.

Aby jeszcze zmniejszyć ilość operacji przesuwania rekordów przy usuwaniu można rozważyć strukturę nieuporządkowanej listy z jednym lub dwo-

ma indeksami. Teraz rekordy rozrzucone są w dowolnych miejscach pamięci, a każdy rekord kończy się dowiązaniem do następnego rekordu (ostatni wskazuje koniec całej struktury). Indeks wskazuje na ulokowanie w pamięci pierwszego rekordu a jeśli używamy dwóch indeksów to i ostatniego.

Rysunek 7.4. Lista z 2 indeksami



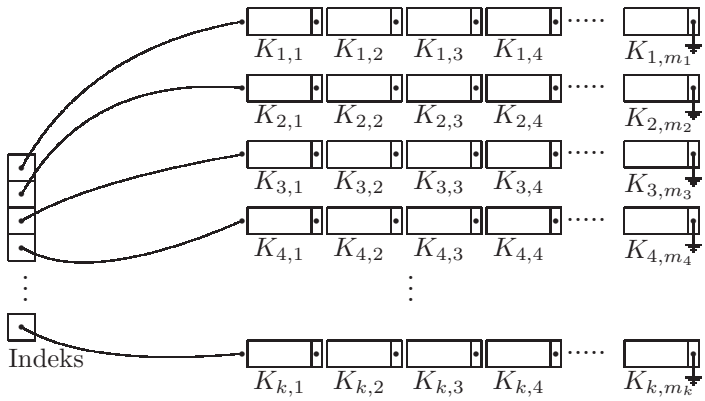
Teraz przy operacji usunięcia, podobnie jak i dodawania nie trzeba przesuwać w pamięci żadnego rekordu, wystarczy tylko zmienić dowiązania. Zatem $A = U = 0$.

7.4.2. Zbiory indeksowo-sekwencyjne i indeksowe

Zbiory indeksowo-sekwencyjne

Struktury te charakteryzują się rozbudowanym indeksem (tablica indeksowa), jednak indeks ten jest mniejszy niż N, dlatego czasem nazywa się tę strukturę indeksem rzadkim. Indeks (tablica) wskazuje sekwencję ułożonych w bloku rekordów Metoda ta jest dość szybka, jeśli chodzi o wyszukiwanie rekordów, jednak dochodzą, czasem bardzo duże koszty utrzymania i przeszukiwania indeksu.

Rysunek 7.5. Zbiory indeksowo - sekwencyjne



Oczywiście, przy przeszukiwaniu bloku $W_{min} = 1, W_{max} = \max_{1\le i\le k} m_i$. Analiza przeciętnego wyszukiwania jest dużo bardziej skomplikowana. Przeanalizujmy najpierw prawdopodobieństwo możliwych rozmieszczeń w k blokach. Ponieważ rozmieszczeń N elementów w k ciągów długości $(m_1, m_2, ...m_k)$, $N = m_1 + m_2 + ... + m_k$, jest $\frac{N!}{m_1!m_2!...m_k!}$, (zob. Twierdzenie 5.1.6) a wszystkich możliwych rozmieszczeń jest k^N (zob. Twierdzenie 5.1.10), więc prawdopodobieństwo układu wynosi

$$p_{m_1,...,m_k} = \frac{N!}{m_1!m_2!...m_k!}\frac{1}{k^N}.$$

Jeżeli mamy układ w poszczególnych blokach $(m_1, m_2, ...m_k), N = m_1 + m_2 + ... + m_k$, to prawdopodobieństwo, że wyszukujemy w i-tym bloku wynosi $\frac{m_i}{N}$ a przeciętna ilośc przeszukanych rekordów w tym bloku, to

$$\sum_{j=1}^{m_i} \frac{j}{m_i} = \frac{m_i + 1}{2},$$

(por. z Przykładem 7.1) a stąd

$$W_{sr} = \frac{1}{k^N}\frac{1}{N}\sum_{i=1}^{k} \sum_{\substack{(m_1,m_2,...m_k)\in\{0,1,..,N\}^k \\ N=m_1+m_2+...+m_k}} \frac{N!}{m_1!m_2!...m_k!}\frac{m_i(m_i+1)}{2}.$$

Upraszczając m_i otrzymujemy

$$W_{sr} = \frac{1}{k^N}\frac{1}{N}\sum_{i=1}^{k} \sum_{\substack{(m_1,m_2,...m_k)\in\{0,1,..,N\}^k \\ N=m_1+m_2+...+m_k}} \frac{N!}{m_1!m_2!...(m_i-2)!m_k!}$$
$$\Big(\frac{m_i+1}{2(m_i-1)}[m_i > 1] + [m_i = 1]\Big),$$

a ponieważ $\frac{1}{2} \leq \frac{l+1}{2(l-1)}[l>1]+[l=1] \leq \frac{3}{2}$, dla wszystkich naturalnych l, i ponieważ

$$\begin{aligned}\sum_{\substack{(m_1,m_2,\ldots m_k)\in\{0,1,..,N\}^k \\ N=m_1+m_2+\ldots+m_k, m_i\geq 2}} & \frac{N!}{m_1!m_2!\ldots(m_i-2)!\ldots m_k!} \\ &= N(N-1)\sum_{\substack{(m_1,m_2,\ldots m_k)\in\{0,1,..,N-2\}^k \\ N-2=m_1+m_2+\ldots+m_k}} \frac{(N-2)!}{m_1!m_2!\ldots(m_i-2)!\ldots m_k!} \\ &= N(N-1)k^{N-2},\end{aligned}$$

więc

$$\frac{1}{2}\frac{N-1}{k} \leq W_{sr} \leq \frac{3}{2}\frac{N-1}{k}.$$

Łatwo widać, że w tym przypadku $U=W$, $A=0$ (nie trzeba nic przesuwać aby dodać rekord), ale dodawanie rekordu wiąże się ze znalezieniem końca bloku, więc ilość przeglądanych rekordów waha się pomiędzy $\min_{1\leq i\leq k} m_i$ a $\max_{1\leq i\leq k} m_i$, identycznie jak wartości $NW_{min}=\min_{1\leq i\leq k} m_i, NW_{max}=\max_{1\leq i\leq k} m_i$. Ilość przeszukiwań przy dodawaniu można zmniejszyć do 1 jeśli będziemy utrzymywać drugą tablicę indeksową wskazującą na końce bloków. Oszacujmy teraz NW_{sr}. Podobnie jak powyżej zachodzi

$$NW_{sr} = \frac{1}{k^N}\frac{1}{N}\sum_{i=1}^{k}\sum_{\substack{(m_1,m_2,\ldots m_k)\in\{0,1,..,N\}^k \\ N=m_1+m_2+\ldots+m_k}} \frac{N!}{m_1!m_2!\ldots(m_i-2)!m_k!}m_i^2,$$

więc

$$\frac{N-1}{k} \leq NW_{sr} \leq \frac{2(N-1)}{k}.$$

W przypadku zbiorów indeksowo-sekwencyjnych dochodzi problem przeszukiwania indeksów. Teraz mamy $W^i_{min}=1, W^i_{max}=k, W^i_{sr}=\sum_{j=1}^{k} j\frac{m_j}{N}$, a więc

$$\frac{(\min_{1\leq j\leq k} m_j)k(k+1)}{2N} \leq W^i_{sr} \leq \frac{(\max_{1\leq j\leq k} m_j)k(k+1)}{2N}.$$

Zbiory indeksowe

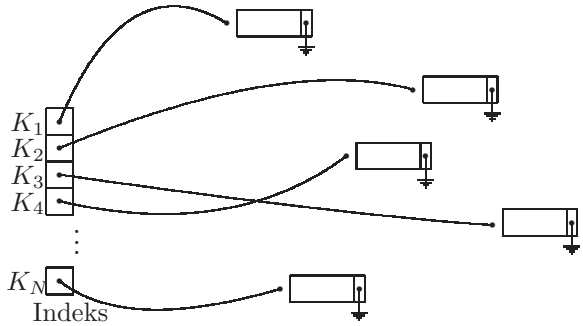
Zbiór nazywamy indeksowym, jeżeli w powyższej konstrukcji $k=N$ a $m_i=1, 1\leq i\leq N$. Tak więc w zbiorze indeksowym każdy rekord ma swój indeks. Analiza tej struktury jest bardzo podobna do analizy zbioru sekwencyjnego, tylko że koszt dotyczący w zbiorze sekwencyjnym rekordów

w zbiorze indeksowym dotyczy tylko i wyłącznie indeksu, rekordy mają tutaj koszt albo 1 (W) albo 0 (U, A). A więc

$$U^i_{min} = W^i_{min} = 1, \;\; U^i_{sr} = W^i_{sr} = \frac{N+1}{2}, \;\; U^i_{max} = W^i_{max} = N.$$

Te dobre wyniki uzyskane są kosztem utrzymywania stosunkowo dużego zbioru indeksów. Zbiory indeksowe czasem się nazywa też indeksowymi gęstymi.

Rysunek 7.6. Zbiory indeksowe

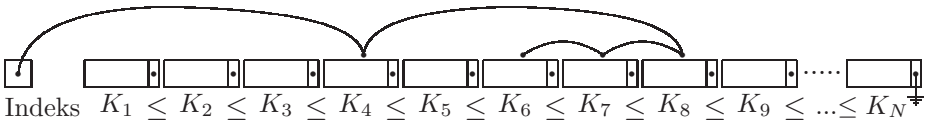


7.4.3. Uporządkowane zbiory sekwencyjne przeszukiwane blokowo lub binarnie

Zakładamy teraz, że właściwy zbiór rekordów uporządkowany jest według rosnącej wartości klucza KLUCZ, według którego poszukujemy rekordy. Najpopularniejsze są wtedy dwie metody wyszukiwania rekordów, które teraz omówimy:

Uporządkowane rekordy przeszukiwane blokowo

Rysunek 7.7. Zbiory uporządkowane sekwencyjne przeszukiwane blokowo



Wyszukiwanie rekordu zawierającego klucz K dla rekordów uporządkowanych blokami długości m polega na kolejnym wyszukiwaniu takiego i, że $K \leq K_{im}$, a po znalezieniu takiego i przesuwamy się wstecz po kluczach $K_{im-1}, K_{im-2}, \cdots, K_{(i-1)m+1}$ aż do znalezienia klucza K. Jeżeli nie

znajdziemy klucza w tym bloku, to klucza K nie ma w przeszukiwanej strukturze. Załóżmy, że m jest dzielnikiem N tak, że $km = N$. Wtedy $W_{min} = 1$ $(K = K_m)$ oraz $W_{max} = k + m - 1$ $(K = K_{(k-1)m+1})$ oraz uwzględniając, że prawdopodobieństwo, iż poszukiwany rekord znajduje się w i-tym bloku $(1 \leq i \leq k)$ wynosi $\frac{1}{k}$ a prawdopodobieństwo, że szukamy w tym bloku i-ty rekord $(1 \leq i \leq m)$ wynosi $\frac{1}{m}$ a więc

$$W_{sr} = \sum_{i=1}^{k} \frac{i}{k} + \sum_{i=0}^{m-1} \frac{i}{m} = \frac{k+1}{2} + \frac{m-1}{2}$$

natomiast $NW = \frac{k+1}{2} + m - 1$.

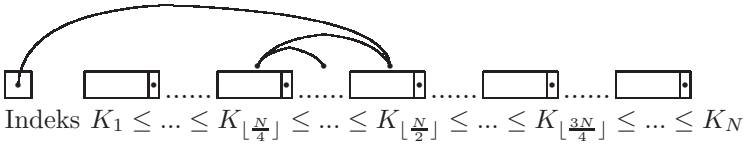
W najlepszym więc przypadku, gdy $m = \lfloor \sqrt{N} \rfloor \approx k$ otrzymujemy $W_{sr} = NW = O(\sqrt{N})$. Jest to wynik porównywalny z wyszukiwaniem w organizacji indeksowo-sekwencyjnej z $k = \lfloor \sqrt{N} \rfloor$, a konieczność utrzymania porządku oznacza duże koszty w przypadku dodawania i usuwania rekordów. Jak łatwo widać

$$A_{min} = U_{min} = 1, A_{sr} = U_{sr} = \frac{N-1}{2}, A_{max} = U_{max} = N.$$

Zdecydowanie lepsze jest następujące rozwiązanie:

Uporządkowane rekordy przeszukiwane binarnie

Rysunek 7.8. Zbiory uporządkowane sekwencyjne przeszukiwane binarnie



Wyszukiwanie wśród binarne uporządkowanych rekordów rekordu zawierającego klucz K polega na podzieleniu szukanego zbioru na dwie (mniej więcej) równe połowy i porównanie klucza K z kluczem rekordu znajdującego się w miejscu podziału $(K_{\lfloor N/2 \rfloor})$. Jeśli $K = K_{\lfloor N/2 \rfloor}$ to poszukiwania kończymy, jeśli $K < K_{\lfloor N/2 \rfloor}$ to przeszukujemy dalej segment $K_1, K_2, ..., K_{\lfloor N/2 \rfloor - 1}$ dzieląc go znowu na połowę, a jeśli $K > K_{\lfloor N/2 \rfloor}$ to dalej przeszukujemy segment $K_{\lfloor N/2 \rfloor + 1}, K_{\lfloor N/2 \rfloor + 2}, ..., K_N$ też dzieląc go na połowę. Oczywiście $W_{min} = 1$. Żeby znaleźć inne wartości wyszukiwań załóżmy, że

$$1 + 2 + 2^2 + ... + 2^{l-1} \leq N < 1 + 2 + 2^2 + ... + 2^l,$$

czyli $2^l \leq N + 1 < 2^{l+1}$ a więc $l = \lfloor \log_2(N+1) \rfloor$. Wtedy, biorąc pod uwagą, że jest $1 = 2^0$ elementów do których docieramy od razy $(K_{\lfloor N/2 \rfloor})$, jest $2 = 2^1$

elementów do których docieramy za 2 razem, ... , 2^i elementów do których docieramy za $i+1$ wyszukiwaniem $(1 \le i \le l-1)$ ..., oraz $(N+1-2^l)$ elementów do których docieramy za $(l+1)$ razem, więc

$$\begin{aligned}
W_{sr} &= \sum_{0\le i<l}(i+1)\frac{2^i}{N} + \frac{N+1-2^l}{N}(l+1) \\
&= \frac{1}{N}\sum_{0\le i<l} i\triangle 2^i + \frac{2^l-1}{N} + \frac{(N+1-2^l)(l+1)}{N} \\
&= \frac{l2^l}{N} - \sum_{0\le i<l}(\triangle i\underline{^1})2^{i+1} + \frac{2^l-1+(N+1-2^l)(l+1)}{N} \\
&= \frac{l2^l+N(l+1)+(1-2^l)l}{N} - \frac{1}{N}\sum_{0\le i<l}2^{i+1} \\
&= \frac{N(l+1)+l}{N} - \frac{2^{l+1}-2}{N} \\
&= \frac{N(l+1)+l-2^{l+1}+2}{N},
\end{aligned}$$

a stąd $\lfloor \log_2(N+1)\rfloor - 1 \le W_{sr} \le \lfloor \log_2(N+1)\rfloor + 1$ a więc $W_{sr} = O(\log_2(N))$ a $NW = l = \lfloor \log_2(N+1)\rfloor = O(\log_2(N))$. Wartości A i U są takie, jak w przypadku rekordów uporządkowanych przeszukiwanych blokowo.

Oba opisane powyżej sposoby organizacji pamięci dobrze się nadają do zbioru danych, w których stosunkowo bardzo często wyszukujemy rekordy, a stosunkowo bardzo rzadko dodajemy lub usuwamy rekordy. Wyszukiwanie binarne jest wyszukiwaniem, które minimalizuje ilość przejrzanych rekordów w najgorszym możliwym przypadku (W_{max}), w stosunku do wszystkich innych znanych algorytmów przeszukiwania uporządkowanych według kluczy rekordów (z wyjątkiem być może zbiorów z dodatkowym indeksem, ale wtedy wyszukiwanie jest przeniesione na indeks).

7.4.4. Odwzorowania $KLUCZ \to ADRES$

Problem 7.4.2. *Czy możliwe jest stworzenie takiej struktury, która miałaby zalety struktury indeksowo-sekwencyjnej lub nawet indeksowej a nie miałaby jej wad, to znaczy nie utrzymywalibyśmy kosztownej struktury indeksowej?* □

Sformułowanie to wydaje się dziwne, jak korzystać z indeksu nie mając indeksu. Jeśli jednak indeks można dla każdego rekordu obliczyć, to nie ma potrzeby utrzymywać skomplikowanych struktur indeksowych w pamięci, obliczenia w komputerze są błyskawiczne. Zwróćmy uwagę, że jedyną możliwą daną, w oparciu o którą możemy wyliczyć ADRES jest KLUCZ, bo

jest to jedyna dana w oparciu o którą chcemy wyszukiwać. Stąd powinniśmy rozważyć możliwe funkcje $h : KLUCZ \to ADRES$. Oczywiście, w praktycznych zastosowaniach przestrzeń możliwych kluczy jest ogromna a obszarem pamięci dysponujemy niewielkim, nie może to więc być funkcja różnowartościowa. Funkcję h nazywamy funkcją haszującą a całą organizację tej struktury tablicą haszującą ("hash table") lub polską nazwą tablicą mieszającą.

Można na haszowanie spojrzeć też inaczej. Pewien obszar pamięci komputera traktuje się jak pokratkowaną kartkę gdzie niektóre kratki mogą być puste a na niektórych mogą być przyczepione pewne rekordy (każdy jeden w jednej kratce) zawierające KLUCZ+DANE. Przy tej organizacji pojawia się kilka problemów.

Po pierwsze co zrobić jeśli trafiły nam się dwa rekordy o tej samej wartości funkcji h to znaczy chcielibyśmy wpisać w tę samą kratkę dwie informacje a na to nie ma miejsca. Problem ten nosi nazwę kolizji a jego rozwiązanie to magazynowanie nadmiarów i można to zrobić na dwa różne sposoby:

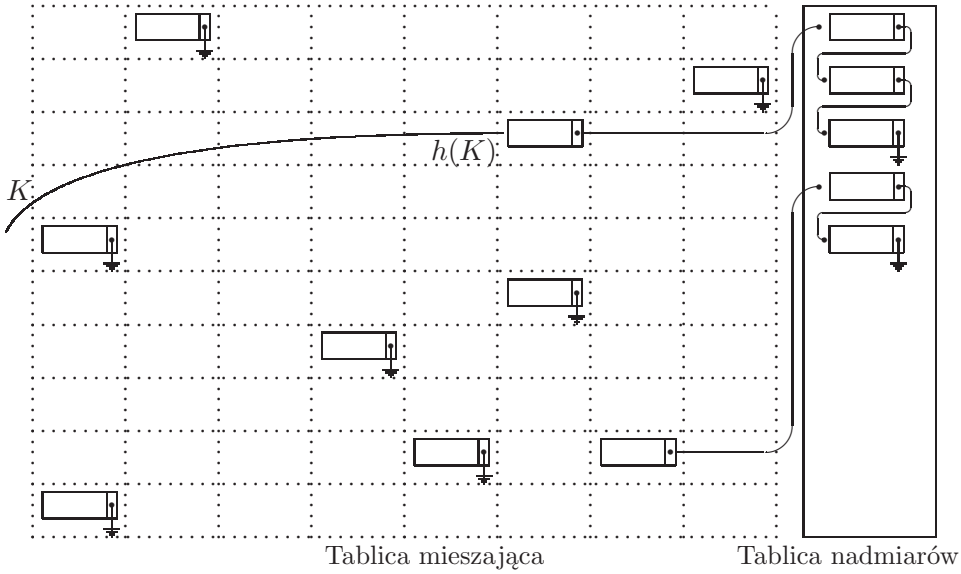
Metoda łańcuchowa - zapisujemy kolizyjne rekordy w odrębnym obszarze pamięci. W kratce wpisujemy jeden rekord i dajemy odsyłacz do wolnej, nie związanej z tablicą haszującą pamięci czyli organizujemy listę.

Metoda otwarta - zapisujemy kolizyjne rekordy w innym miejscu tablicy haszującej.

- **Adresowanie liniowe** Wpisujemy drugi, kolizyjny, rekord w następne wolne miejsce w tabeli.
- **Adresowanie kwadratowe** Wpisujemy drugi rekord do pozycji 1 -następnej, 1 - poprzedniej, 2^2 następnej, 2^2 poprzedniej, 3^2 następnej itp. (dopóki nie znajdziemy wolnej)
- **Rehaszowanie** Używamy drugiej funkcji haszującej do wyznaczenia następnego miejsca dla kolizyjnego rekordu.

Drugim problemem jest wybór funkcji haszującej h. Funkcja ta powinna być wybrana tak, aby rozkład rekordów w tablicy był możliwie równomierny. Jednak nie zawsze potrafimy przewidzieć jakie rekordy pojawią się w tabeli a ponadto same klucze mogą nie pojawiać się niezależnie, a funkcję haszującą powinniśmy wybrać na początku. Dlatego istnieje pewne niewielkie prawdopodobieństwo, że po wyborze funkcji haszującej i po wpisaniu danych okazało się, że wszystkie rekordy (poza jednym) są nadmiarem, jest to koszmar metody haszowania. Najważniejszym więc wymaganiem wobec funkcji haszującej jest brak prostych zależności między obiektem a wartością jego "hasha". Tak więc przyjęcie za funkcję haszującą wartości pierwszej lub ostatniej cyfry będzie skutkować fatalną wydajnością struktur danych, które z nich korzystają.

Rysunek 7.9. Tablice mieszające ("hash table")



Czasem zamiast pełnych rekordów podaje się we właściwej tablicy mieszającej same odsyłacze, a wszystkie rekordy są umiejscowione w listach (lub zorganizowanych sekwencyjnie zbiorach) w tablicy nadmiarów. Podajmy teraz przykład takiej metody haszowania.

Przykład 7.4.1. *Interesują nas dane dotyczące osób (np. studentów) w grupie. Żeby nie komplikować przyjmijmy, że dobrym identyfikatorem jest imię studenta (nie ma dwóch osób o tym samym imieniu) a funkcja haszująca jest oparta na tym, jaka jest pierwsza litera imienia:*

$$h(IMIĘ) = \begin{cases} 1, & dla \quad A - F, \\ 2 & dla \quad G - L, \\ 3 & dla \quad M - R,, \\ 4 & dla \quad S - Z, \end{cases}$$

Tak więc $h(ANNA) = 1$, $h(TOMASZ) = 4$, $h(JAN) = 2$. *Tablica mieszająca składa się więc tylko z 4 kratek (zaiste to supermała tablica). Załóżmy, że w tablicy mieszającej znajdują sie same KLUCZE a odsyłacz do miejsc w pamięci gdzie te dane są opisane jest w tablicy DANE[], jest to więc też struktura indeksowa. Załóżmy, że nadmiar jest zorganizowany w pamięci komputera w postaci listy którą realizują dwie tablice FIRST[] i NEXT[]. Jeśli więc do tablicy wpisywane są kolejno osoby ANNA, TOMASZ, DIANA,*

AGATA, JOANNA to zmienna $n = 5$ a w efekcie wstawiania otrzymamy następujące struktury:

Nr	*KLUCZ*	*DANE*	*NEXT*
1	*ANNA*	*"dane ANNY"*	*3*
2	*TOMASZ*	*"dane TOMASZA"*	*0*
3	*DIANA*	*"dane DIANY"*	*4*
4	*AGATA*	*"dane AGATY"*	*0*
5	*JOANNA*	*"dane JOANNY"*	*0*

oraz

FIRST[1]=1	*FIRST[2]=5*	*FIRST[3]=-1*	*FIRST[4]=2*
1. ANNA 3	*5. JOANNA 0*		*2. TOMASZ 0*
3. DIANA 4			
4. AGATA 0			

Jak widać pusta lista (puste miejsce w tabeli haszującej) oznaczone jest w FIRST[] przez -1 a koniec listy w NEXT[] przez 0. Algorytm wyszukiwania i ewentualnego dopisywania klucza K mógłby teraz wyglądać następująco:

Algorytm 7.7. Algorytm **Tablice haszujące**

A1. Podstaw i:=h(K) oraz j:=FIRST[i]
A2. Jeśli j≤0 to rekordu nie znaleziono.
Jeśli chcesz ten rekord dopisać do bazy to przejdź do kroku A5
w przeciwnym razie STOP.
A3. Jeśli KEY[i]=K to wydaj DATA[i] i STOP
(Klucz znaleziono, jest to i-ty element)
A4. Podstaw i:=j, potem j:=NEXT[i] i wróć do A2.
A5. Podstaw n:=n+1
A6. Jeśli j<0 to FIRST[i]:=n w przeciwnym razie NEXT[i]:=n
A7. Podstaw KEY[n]:=K, DANE[n]="dane
klucza K", NEXT[n]:=0 i STOP.

Podstawowym zadaniem programisty przy stosowaniu metod haszujących jest wybór funkcji haszującej h. Jak wspomnieliśmy, powinna być ona taka, aby rozkład wszystkich możliwych danych był równomierny. W powyższym przykładzie należałoby na przkład wziąść losowy fragment książki telefonicznej i przeanalizować czy jest tyle samo imion zaczynających się na A-F, G-L, M-R, S-Z. Wydaje się że funkcja haszująca użyta w tym przykładzie nie jest najlepsza (zdecydowanie bardzo dużo jest imion żeńskich na A).

Wykonajmy teraz analizę (probabilistycznym) średniej liczby porównań algorytmu. Załóżmy, że N kluczy w tablicy jest idealnie równomiernie rozłożone i że tablica ma wymiar m. Rozważmy dwie sytuacje

• *Gdy wyszukiwanego klucza K nie ma w tablicy. Wtedy liczba porównań potrzebnych do stwierdzenia tego faktu wynosi*

$$V = I[h(KEY[1]) = h(K)]+I[h(KEY[2]) = h(K)]+...+I[h(KEY[N]) = h(K)]$$

gdzie $I[A]$ jest to indykator zdarzenia A. Ze względu na nasze założenia mamy w V sumę niezależnych zmiennych losowych o tym samym rozkładzie $P[h(KEY[i]) = h(K)] = \frac{1}{m}$, a stąd

$$NW = EV = \frac{N}{m}.$$

• *Załóżmy teraz, że klucz K znaleziono, jest to klucz o numerze L. Teraz, podobnie jak poprzednio, L jest zmienną losową o rozkładzie jednostajnym na zbiorze $\{1, 2, 3, ..., N\}$ to znaczy $P[L = i] = \frac{1}{N}$. Wtedy liczba porównań potrzebnych do znalezienia L wynosi*

$$V = I[h(KEY[1]) = h(K)]+I[h(KEY[2]) = h(K)]+...+I[h(KEY[L]) = h(K)].$$

Jeśli zmienna losowa L nie zależy od rozkładu kluczy w tablicy, to

$$W_{sr} = EV = \frac{1}{m}EL = \frac{1}{m}\sum_{i=1}^{N} i\frac{1}{N} = \frac{N+1}{2m}.$$

Zawsze więc ilość porównań jest rzędu $\frac{N}{m}$. Jeśli teraz N jest bardzo duże a $m = \frac{N}{\ln(N)}$, to w przybliżeniu przeciętnie wystarcza tylko $O(\ln(N))$ porównań, tak jak w wyszukiwaniu binarnym sekwencyjnie uporządkowanych rekordów (podrozdz. 7.5.3), a jeśli m jest porównywalne z N to może wystarczyć tylko kilka porównań. Jest to zdecydowanie lepszy wynik niż w poprzednio rozważanych strukturach.

Oczywiście $W_{max} = N$ i jest to koszmar, który na szczęście ma bardzo niewielkie prawdopodobieństwo zajścia. □

Zakończmy ten podrozdział podając przykłady prostych funkcji haszujących:

Dzielenie Indeks wyznacza się jako resztę z dzielenia klucza K przez rozmiar tablicy m:

$$h(K) = K \bmod m.$$

Jeśli m nie jest liczbą pierwszą to jest lepiej użyć funkcji: $h(K) = (K \bmod p) \bmod m$ gdzie: p jest liczbą pierwszą i $p > m$. Gdyby wybrać jako m liczbę złożoną, na przykład w skrajnym przypadku $m = 2^k$, to wtedy $h(K)$ byłaby po prostu wartością najmniej znaczących k bitów klucza K, a wtedy, do prawidłowego używania takiej funkcji haszującej, musielibyśmy dorzucić założenie (niekoniecznie prawdziwe i weryfikowalne) równomiernego rozkładu tych k bitów.

Składanie z przesuwaniem Indeks wyznacza się dzieląc klucz na fragmenty, a następnie je dodając, np.

$$h(458372) = 458 + 37 + 2 = 497, \text{ lub } h(458372) = 45 + 83 + 72 = 200.$$

Składanie brzegami Indeks wyznacza się dzieląc klucz na fragmenty, a następnie dodając je zmieniając kolejność cyfr w co drugim fragmencie, np.

$$h(458372) = h(458 + 37 + 2) = 458 + 73 + 2 = 533,$$

lub

$$h(458372) = h(45 + 83 + 72) = 45 + 38 + 72 = 155.$$

Środek kwadratu Indeks wyznacza się podnosząc klucz do kwadratu, a następnie wybierając środkowy fragment wyniku, np.

$$h(458372] = F((458372)^2) = F(2101\boxed{0489}0384) = 489,$$

gdzie $F()$ oznacza operację wycinania.

Wycinanie Indeks wyznacza się wybierając dowolny fragment klucza, np.

$$h(45\boxed{837}2) = 837, \text{ lub } h(458\boxed{372}) = 372.$$

Zamiana podstawy Indeks wyznacza się zmieniając podstawę kodu klucza (zob. podrozdz. 3.5), np.

$$h((458372)_{10}) = h((1577204)_8) = 1577204,$$

lub

$$h((458372)_{10}) = h((154131442)_5) = 154131442.$$

Ćwiczenie 7.4.1. *Załóżmy, że mamy następujący zbiór danych (wraz z innymi dodatkowymi informacjami o tych osobach)*

Identyfikator	Imię i Nazwisko
134912	Jan Wiśniewski
350334	Sylwia Nowak
383321	Tomasz Kowal
134055	Elżbieta Wójcik
256722	Alina Majewska
200912	Kamil Kłosowski
267021	Robert Szuster
113451	Józef Kowalski
709942	Anna Białecka
334305	Franciszek Nurowski
995565	Diana Politkowska

gdzie identyfikator K jest zawsze sześciocyfrowym ciągiem $K = (c_1c_2c_3c_4c_5c_6)_{10}$. Która, z następujących funkcji haszujących jest najlepsza dla rozmieszczenia tych danych w tablicy haszującej o rozmiarze $m = 19$:

$$\begin{aligned} h_1(K) &= K \bmod 19, \\ h_2(K) &= ((c_1c_2)_{10} + (c_3c_4)_{10} + (c_5c_6)_{10}) \bmod 19, \\ h_3(K) &= ((c_1c_2)_{10} + (c_4c_3)_{10} + (c_5c_6)_{10}) \bmod 19, \\ h_4(K) &= (c_3c_4)_{10} \bmod 19, \\ h_5(K) &= h_5((d_0d_1...d_n)_8) = (\sum_{k=0}^{n} d_k 10^{n-k}) \bmod 19? \end{aligned}$$

Odpowiedź uzasadnij. □

Podsumowanie

- W rozdziale tym rozwinęliśmy jeszcze dwa elementy potrzebne do poprawnej analizy problemów informatycznych. Były to podstawy rachunku prawdopodobieństwa stanowiącego naturalne rozwinięcie rozdziału 5 - kombinatoryki oraz analizę asymptotycznego zachowania ciągów.
- W podrozdziale 7.3 analizowaliśmy grupę algorytmów porządkowania ciągu (algorytmów sortowania). Wyniki tej analizy podsumujemy w ta-

belce:

Algorytm	Ilość porównań			Ilość wymian		
	min	r	max	min	r	max
proste wstawianie	$O(n)$	$O(n^2)$	$O(n^2)$	$O(n)$	$O(n^2)$	$O(n^2)$
wstawianie połówkowe	$O(n)$	$O(n\ln(n))$	$O(n^2)$	$O(n)$	$O(n^2)$	$O(n^2)$
proste wybieranie	$O(n^2)$	$O(n^2)$	$O(n^2)$	$O(n)$	$O(n\ln(n))$	$O(n^2)$
sortowanie bąbelkowe	$O(n^2)$	$O(n^2)$	$O(n^2)$	$o(1)$	$O(n^2)$	$O(n^2)$
sortowanie mieszane	$O(n^2)$	$O(n^2)$	$O(n^2)$	$o(1)$	$O(n)$	$O(n^2)$
quickosort z $rand()$	$O(n\ln(n))$	$O(n\ln(n))$	$O(n^2)$	$O(n\ln(n))$	$O(n\ln(n))$	$O(n^2)$

- w podrozdziale 7.4 rozważaliśmy problem rozmieszczenia rekordów w pamięci komputera, szczególnie istotny w teorii baz danych. Analizę podsumujemy w tabelce.

Organizacja	W_{min}	W_{sr}	W_{max}	NW
Sekwencyjna	$O(1)$	$O(N)$	$O(N)$	$O(N)$
Indeksowo-sekwencyjna	$O(1)$	$O(\frac{N}{k})^{ab}$	$O(\frac{N}{k})^{ab}$	$O(\frac{N}{k})^{ab}$
Indeksowa	$O(1)$	$O(1)$	$O(1)$	$O(1)$
Uporządkowane przesz. blokowe	$O(1)$	$O(k+\frac{N}{k})^{ab}$	$O(k+\frac{N}{k})^{ab}$	$O(k+\frac{N}{k})^{ab}$
Uporządkowane przesz. binarnie	$O(1)$	$O(\ln(N))$	$O(\ln(N))$	$O(\ln(N))$
Tablice mieszające	$O(1)$	$O(\frac{N}{m})^{cd}$	$O(N)$	$O(\frac{N}{m})^{cd}$

[a] k jest ilością bloków.
[b] Minimalnie $O(\sqrt{N})$
[c] m jest wielkością tablicy mieszającej.
[d] Minimalnie $O(1)$

Rozdział 8

Grafy

8.1. Podstawowe pojęcia i problem reprezentacji **154**
8.2. Grafy nieskierowane **158**
8.3. Grafy skierowane **177**
8.4. Drzewa **179**
8.5. Zastosowania teorii grafów **188**

W rozdziale tym wprowadzimy podstawowe pojęcia i omówimy podstawowe problemy związane z teorią grafów.

8.1. Podstawowe pojęcia i problem reprezentacji

8.1.1. Definicje

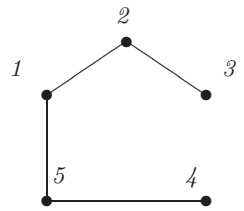
Definicja 8.1.1. *Grafem nieskierowanym $\mathcal{G}$ nazywamy uporządkowaną parę (V, E) gdzie V to pewien skończony (albo nieskończony) zbiór zwany zbiorem wierzchołków (węzłów) grafu a $E \subset \{< \{v_1, v_2\}, k >: v_1, v_2 \in V, k \in \mathcal{N}\backslash\{0\}\}$, zbiór dwuelementowych podzbiorów zbioru V zwany zbiorem krawędzi grafu, trzecim elementem jest krotność krawędzi (dwa wierzchołki mogą być połączone jedną, dwoma, trzema,... krawędziami). Graf $\mathcal{G}$ nazywamy grafem bez krawędzi wielokrotnych, jeśli w każdej trójce $< \{v_1, v_2\}, k > \in E(\mathcal{G})$ mamy $k = 1$. Piszemy wtedy w krótszej formie $E(\mathcal{G}) \in \{\{v_1, v_2\}, v_1, v_2 \in V(\mathcal{G})\}$ po prostu pomijając ostatni element k $(= 1)$.* □

Zauważmy, że $< \{v_1, v_2\}, k > = < \{v_2, v_1\}, k >$ gdyż kolejność podawania elementów zbioru jest nieistotna.

Definicja 8.1.2. *Grafem skierowanym $\mathcal{G}$ nazywamy uporządkowaną parę (V, E) gdzie V pewien skończony (albo nieskończony) zbiór zwany zbiorem wierzchołków (węzłów) grafu i $E \in V \times V \times (\mathcal{N}\backslash\{0\})$, zbiór uporządkowanych trójek zbioru V, gdzie pierwszym elementem jest początek, drugim koniec krawędzi a trzecim jest krotność krawędzi. Graf $\mathcal{G}$ nazywamy grafem bez krawędzi wielokrotnych, jeśli w każdej trójce $< v_1, v_2, k > \in E(\mathcal{G})$ mamy $k = 1$. Piszemy wtedy krótko $E(\mathcal{G}) \in \{< v_1, v_2 >, v_1, v_2 \in V(\mathcal{G})\}$.* □

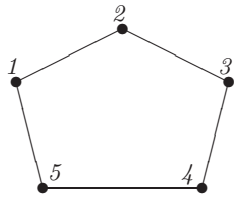
Często piszemy $V(\mathcal{G})$ zamiast V oraz $E(\mathcal{G})$ zamiast E. Stąd $\mathcal{G}$=$(V(\mathcal{G}), E(\mathcal{G}))$.

Przykład 8.1.1. *Niech $V(\mathcal{G}) = \{1, 2, 3, 4, 5\}, E(\mathcal{G}) = \{\{1, 2\}, \{2, 3\}, \{1, 5\}, \{4, 5\}\}$. Uporządkowany ciąg krawędzi $< \{4, 5\}, \{5, 1\}, \{1, 2\}, \{2, 3\} >$ nazywamy drogą.*



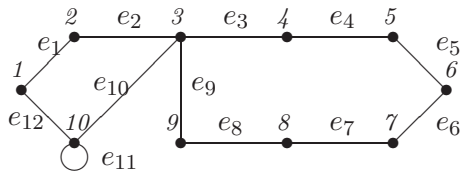
□

Przykład 8.1.2. *Niech $V(\mathcal{G}) = \{1, 2, 3, 4, 5\}, E(\mathcal{G}) = \{\{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\}\}$. Drogę $< \{1, 2\}, \{2, 3\}, \{3, 4\}, \{4, 5\}, \{5, 1\} >$ nazywamy cyklem (cykl jest szczególnym rodzajem drogi).*



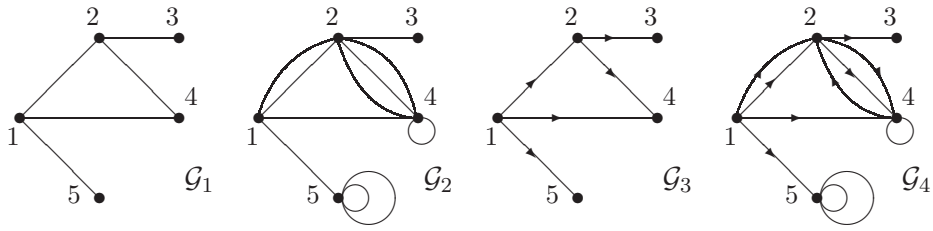
□

Definicja 8.1.3. *Drogą o długości n nazywamy ciąg krawędzi* $e_1e_2e_3...e_n$ *taki, że* $e_i \cap e_{i+1} = \emptyset, i = 1, 2, 3, ..., n-1$. *Jeśli* $e_1 \cap e_n \neq \emptyset$ *to taką drogę nazywamy zamkniętą. Jeśli* $e_i \neq e_j$ *dla* $i \neq j$ *to taką drogę nazywamy prostą.*



Krawędź której początkowy i końcowy wierzchołek jest taki sam (np. $e_{11} = \{10, 10\}$*) nazywamy pętlą. Jeśli na drodze nie ma pętli to taką drogę nazywamy ścieżką. Jeśli droga jest zamknięta i nie zawiera pętli to nazywamy ją cyklem i oznaczamy* C_n. *Jeśli graf* $\mathcal{G}$ *nie zawiera cykli to nazywamy go acyklicznym. Graf bez pętli i krawędzi wielokrotnych nazywamy grafem prostym.* □

Rysunek 8.1. Różne typy grafów



Graf $\mathcal{G}_1$ $(V = \{1, 2, 3, 4, 5\}, E = \{\{1, 2\}, \{1, 4\}, \{1, 5\}, \{2, 3\}, \{2, 4\}\})$ jest grafem prostym (bez pętli i krawędzi wielokrotnych) nieskierowanym. Graf ma cykl $\{\{1, 2\}, \{2, 4\}, \{1, 4\}\}$.

Graf $\mathcal{G}_2$ $(V = \{1, 2, 3, 4, 5\}, E = \{< \{1, 2\}, 2 >, < \{1, 4\}, 1 >, < \{1, 5\}, 1 >$, $< \{2, 3\}, 1 >, < \{2, 4\}, 3 >, < \{4, 4\}, 1 >, < \{5, 5\}, 2 >\})$ nie jest grafem prostym, ma pętlę dwukrotną przy wierzchołku 5 i jedną pętlę przy wierzchołku 4, krawędź $\{2, 4\}$ trzykrotną a krawędź $\{1, 2\}$ dwukrotną. Graf ma cykl $\{\{1, 2\}, \{2, 4\}, \{4, 1\}\}$.

Graf $\mathcal{G}_3$ $(V = \{1,2,3,4,5\}, E = \{<1,2>, <1,4>, <1,5>, <2,3>, <2,4>\})$ jest grafem prostym skierowanym. Graf jest acykliczny.

Graf $\mathcal{G}_4$ $(V = \{1,2,3,4,5\}, E = \{<1,2,2>, <1,4,1>, <1,5,1>, <2,3,1>, <2,4,2>, <4,2,1>, <4,4,1>, <5,5,2>\})$ nie jest grafem prostym, jest grafem skierowanym. Graf jest acykliczny.

Twierdzenie 8.1.1. *Jeśli $\mathcal{G}$ jest grafem acyklicznym, to dla każdej pary wierzchołków $u, v \in V(\mathcal{G})$ istnieje co najwyżej jedna droga prowadząca od u do v.*

Dowód. Jeśli dwie takie drogi (d_1 i d_2) istnieją, to musi być co najmniej na jednej drodze wierzchołek $r \in d_1$ który nie istnieje na drodze d_2 (ewentualnie zmieniamy nazwy dróg). Przechodząc po drodze d_1 w przód i w tył od wierzchołka r dopóty dopóki nie napotkamy na wierzchołek z drogi d_2 (że musimy napotkać wynika z faktu, że $u, v \in d_1 \wedge u, v \in d_2$.) Pierwsze napotkane takie wierzchołki wraz z fragmentem drogi d_2 tworzą cykl. □

8.1.2. Implementacja

Graf $\mathcal{G}$ może być reprezentowany albo bezpośrednio, poprzez podanie elementów zbiorów $V(\mathcal{G})$ oraz $E(\mathcal{G})$. Różne metody implementacji grafów prześledzimy na przykładzie czterech grafów z Rysunku 8.1.

Macierz sąsiedztwa $M_{\mathcal{G}} = [m_{i,j}]$. Macierz ta jest rozmiaru $\overline{\overline{V(\mathcal{G})} \times \overline{V(\mathcal{G})}}$ i $m_{i,j}$ oznacza ilość krawędzi grafu $\mathcal{G}$ łączących i-ty i j-ty wierzchołek. Pętle wierzchołka i zaznaczamy w elemencie $m_{i,i}$. Jeżeli graf jest nieskierowany to macierz sąsiedztwa jest symetryczna. W przypadku grafu, którego krawędzie mają wagi, gdy nie dopuszczamy krawędzi wielokrotnych w macierzy sąsiedztwa mogą wystąpić wagi krawędzi. Wielkość pamięci tej struktury to $O(\overline{\overline{V}}^2)$.

Tabela 8.1. Reprezentacja grafów za pomocą macierzy sąsiedztwa

$$M_{\mathcal{G}_1} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad M_{\mathcal{G}_2} = \begin{bmatrix} 0 & 2 & 0 & 1 & 1 \\ 2 & 0 & 1 & 3 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 3 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 2 \end{bmatrix},$$

$$M_{\mathcal{G}_3} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad M_{\mathcal{G}_4} = \begin{bmatrix} 0 & 2 & 0 & 1 & 1 \\ 0 & 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix},$$

Lista incydencji $L_{\mathcal{G}}$. Należy utworzyć listy dla każdego wierzchołka v, w której przechowujemy zbiór wierzchołków połączonych krawędzią z v. Dla grafu skierowanego podajemy listę tych wierzchołków, które są końcem krawędzi, gdy początkiem jest v. W przypadku grafów z wagami krawędzi lista poza wierzchołkiem powinna również zawierać wagę. Wielkość zajętej pamięci w tej metodzie to $O(\overline{\overline{V}} + \overline{\overline{E}})$.

Tabela 8.2. Reprezentacja grafu za pomocą listy incydencji

$L_{\mathcal{G}_1}$	$L_{\mathcal{G}_2}$	$L_{\mathcal{G}_3}$	$L_{\mathcal{G}_4}$
1 : 2, 4, 5;	**1** : 2, 2, 4, 5;	**1** : 2, 4, 5;	**1** : 2, 2, 4, 5;
2 : 1, 3, 4;	**2** : 1, 1, 3, 4, 4, 4;	**2** : 3, 4;	**2** : 3, 4, 4;
3 : 2;	**3** : 2;	**3** :;	**3** :;
4 : 1, 2;	**4** : 1, 2, 2, 2, 4;	**4** :;	**4** : 2, 4;
5 : 1;	**5** : 1, 5, 5;	**5** :;	**5** : 5, 5;

Lista krawędzi $I_{\mathcal{G}}$. Jest to w zasadzie bezpośrednia reprezentacja $E(\mathcal{G})$, lista na której bezpośrednio podajemy wszystkie krawędzie grafu. Koszt to $O(\overline{\overline{E}})$.

Tabela 8.3. Reprezentacja grafu za pomocą listy krawędzi

$I_{\mathcal{G}_1}$	$I_{\mathcal{G}_2}$	$I_{\mathcal{G}_3}$	$I_{\mathcal{G}_4}$
$(1,2)(2,1)$	$(1,2)(2,1)$	$(1,2)(2,3)$	$(1,2)(1,2)$
$(1,4)(4,1)$	$(1,2)(2,1)$	$(1,4)(1,5)$	$(1,4)(1,5)$
$(1,5)(5,1)$	$(1,4)(1,5)$	$(2,4)$	$(2,3)(2,4)$
$(2,3)(3,2)$	$(2,3)(3,2)$		$(2,4)(4,2)$
$(2,4)(4,2)$	$(2,4)(4,2)$		$(4,4)(5,5)$
	$(2,4)(4,2)$		$(5,5)$
	$(2,4)(4,2)$		
	$(4,4)(5,5)$		
	$(5,5)$		

Macierz incydencji $\Upsilon = [\mu_{i,j}]$. to tablica o rozmiarach $\overline{\overline{E}} \times \overline{\overline{V}}$. Składa się ona z $\overline{\overline{E}}$ wierszy i $\overline{\overline{V}}$ kolumn. Jeśli krawędź wychodzi z danego wierzchołka to piszemy w odpowiedniej kolumnie (-1), jeśli do niego wchodzi piszemy $(+1)$, jeśli wierzchołek nie należy do krawędzi piszemy 0 a jeśli jest to pętla własna piszemy 2. Koszt tej reprezentacji to $O(\overline{\overline{V}} \cdot \overline{\overline{E}})$.

$$\mu_{i,j} = \begin{cases} 0 & \text{jeśli } v_i \text{ nie jest punktem końcowym } e_j \\ -1 & \text{jeśli } v_i \text{ jest punktem początkowym } e_j \\ 1 & \text{jeśli } v_i \text{ nie jest punktem końcowym } e_j \\ 2 & \text{jeśli } e_j \text{ jest pętlą wierzchołka } v_i \end{cases}$$

Gdyby krawędzie miały swoje wagi, to można by je umieszczać zamiast liczb ± 1 lub dorzucić jeszcze jedną kolumnę z wagami.

Tabela 8.4. Reprezentacja grafu za pomocą macierzy incydencji

$$\Upsilon_{\mathcal{G}_1} = \left[\begin{array}{c|ccccc} E & 1 & 2 & 3 & 4 & 5 \\ \hline 1-2 & 1 & 1 & 0 & 0 & 0 \\ 1-4 & 1 & 0 & 0 & 1 & 0 \\ 1-5 & 1 & 0 & 0 & 0 & 1 \\ 2-3 & 0 & 1 & 1 & 0 & 0 \\ 2-4 & 0 & 1 & 0 & 1 & 0 \end{array}\right], \qquad \Upsilon_{\mathcal{G}_2} = \left[\begin{array}{c|c|ccccc} E & \text{waga} & 1 & 2 & 3 & 4 & 5 \\ \hline 1-2 & 2 & 1 & 1 & 0 & 0 & 0 \\ 1-4 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1-5 & 1 & 1 & 0 & 0 & 0 & 1 \\ 2-3 & 1 & 0 & 1 & 1 & 0 & 0 \\ 2-4 & 3 & 0 & 1 & 0 & 1 & 0 \\ 4-4 & 1 & 0 & 0 & 0 & 2 & 0 \\ 5-5 & 2 & 0 & 0 & 0 & 0 & 2 \end{array}\right],$$

$$\Upsilon_{\mathcal{G}_3} = \left[\begin{array}{c|ccccc} E & 1 & 2 & 3 & 4 & 5 \\ \hline 1-2 & -1 & 1 & 0 & 0 & 0 \\ 1-4 & -1 & 0 & 0 & 1 & 0 \\ 1-5 & -1 & 0 & 0 & 0 & 1 \\ 2-3 & 0 & -1 & 1 & 0 & 0 \\ 2-4 & 0 & -1 & 0 & 1 & 0 \end{array}\right], \qquad \Upsilon_{\mathcal{G}_4} = \left[\begin{array}{c|c|ccccc} E & \text{waga} & 1 & 2 & 3 & 4 & 5 \\ \hline 1-2 & 2 & -1 & 1 & 0 & 0 & 0 \\ 1-4 & 1 & -1 & 0 & 0 & 1 & 0 \\ 1-5 & 1 & -1 & 0 & 0 & 0 & 1 \\ 2-3 & 1 & 0 & 1 & 1 & 0 & 0 \\ 2-4 & 3 & 0 & -1 & 0 & 1 & 0 \\ 4-4 & 1 & 0 & 0 & 0 & 2 & 0 \\ 5-5 & 2 & 0 & 0 & 0 & 0 & 2 \end{array}\right].$$

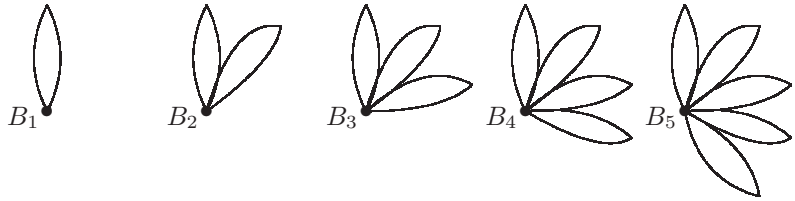
8.2. Grafy nieskierowane

8.2.1. Niektóre klasy grafów nieskierowanych

Graf pusty $\mathcal{G}_\emptyset = (\emptyset, \emptyset)$ to graf, którego krawędzie i wierzchołki są zbiorami pustymi.

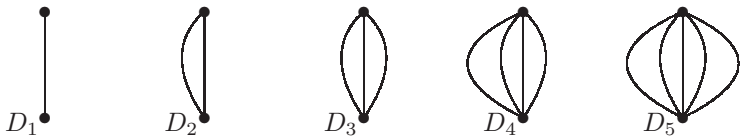
Bukiet B_n jest grafem zawierającym jeden wierzchołek i n pętli w tym wierzchołku.

Rysunek 8.2. Bukiet B_n



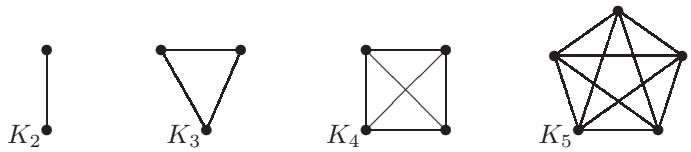
Dipol D_n jest grafem zawierającym dwa wierzchołki i n krawędzi łączących te wierzchołki.

Rysunek 8.3. Dipol D_n



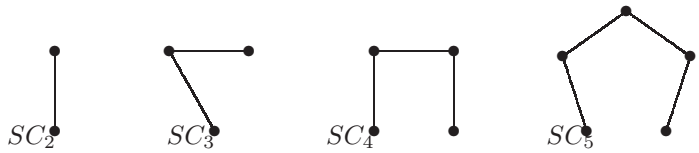
Graf kompletny K_n to graf składający się z n wierzchołków i krawędzi łączących każde dwa wierzchołki. K_n nie ma pętli ani krawędzi wielokrotnych.

Rysunek 8.4. Graf kompletny K_n



Graf n-ścieżka SC_n to graf zawierający n wierzchołków $V(\mathcal{SC}_n) = \{v_1, v_2, ..., v_n\}$ i dokładnie $n-1$ krawędzi $E(\mathcal{SC}_n) = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, ..., \{v_{n-1}, v_n\}\}$.

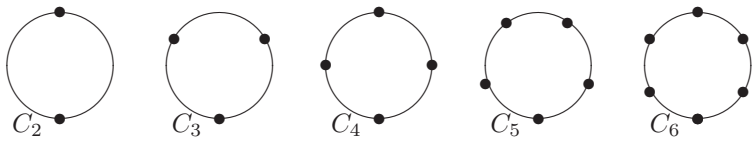
Rysunek 8.5. n-ścieżka SC_n



Graf n-cykl C_n to graf zawierający n wierzchołków $V(\mathcal{S}_n) = \{v_1, v_2, ..., v_n\}$

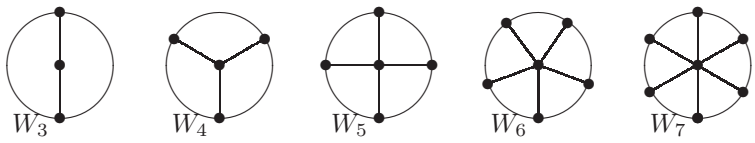
i dokładnie n krawędzi $E(\mathcal{S}_n) = \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_3, v_4\}, ..., \{v_{n-1}, v_n\}, \{v_n, v_1\}\}$.

Rysunek 8.6. n-cykl C_n



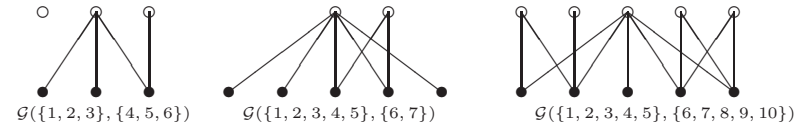
Graf n-koło W_n to C_{n-1} cykl z wierzchołkami $V(C_{n-1}) = \{v_1, v_2, ..., v_{n-1}\}$ z dodanym jednym wierzchołkiem v_n, który ma krawędzie łączące z każdym z z wierzchołków $V(C_{n-1})$.

Rysunek 8.7. n-koło W_n



Graf dwudzielny $\mathcal{G}(V_1, V_2)$ jest to graf, którego zbiór wierzchołków można podzielić na dwa podzbiory takie, że żadne dwa wierzchołki z tego samego podzbioru nie są połączone krawędzią.

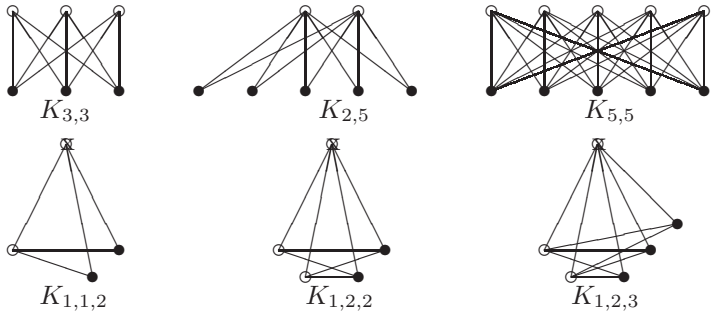
Rysunek 8.8. Graf dwudzielny $\mathcal{G}(V_1, V_2)$



Pełny graf dwudzielny K_{r_1,r_2} jest to graf, którego zbiór wierzchołków można podzielić na dwa podzbiory liczebności odpowiednio r_1 oraz r_2 takie, że żadne dwa wierzchołki z tego samego podzbioru nie są połączone krawędzią a zbiór krawędzi zawiera wszystkie krawędzie łączą-

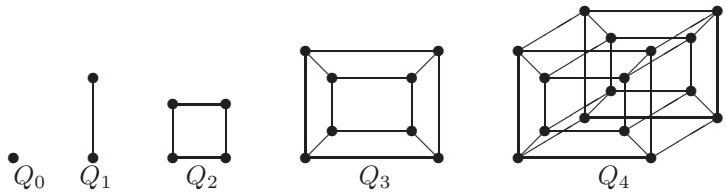
ce wierzchołki jednego podzbioru z wierzchołkami drugiego podzbioru. Analogicznie definiujemy grafy trójdzielne K_{r_1,r_2,r_3} itd.

Rysunek 8.9. Pełne grafy dwudzielne, trójdzielne ...



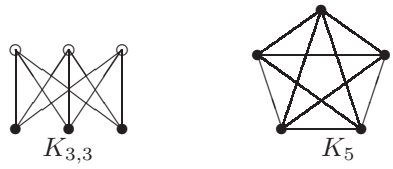
d-wymiarowy hipersześcian Q_d to graf zawierający 2^d wierzchołków, które etykietujemy ciągami binarnymi długości d tak, że żadne dwa różne wierzchołki nie mają tej samej etykiety, a krawędzie łączą wierzchołki, których ciągi binarne (etykiety) różnią się na dokładnie jednej pozycji. Taki sposób etykietowania nazywamy kodami Gray'a. Szerzej o tych grafach i kodach Gray'a piszemy w podrozdziale 8.2.5.

Rysunek 8.10. Grafy hipersześcienne Q_n



Grafy Kuratowskiego to grafy K_5 i $K_{3,3}$. Szerzej o znaczeniu tych grafów w podrozdziale 8.2.4.

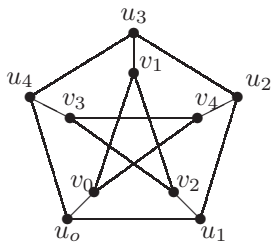
Rysunek 8.11. Grafy Kuratowskiego



Graf Petersena *Pet* jest to graf skonstruowany z dwóch rozłącznych 5-cykli

$\{u_0, u_1, u_2, u_3, u_4\}$ oraz $\{v_0, v_1, v_2, v_3, v_4\}$ połączone miedzy sobą krawędziami $\{u_i, v_{2i \bmod 5}\}, i = 0, 1, 2, 3, 4\}$.

Rysunek 8.12. Graf Petersena



Pet

Grafy $\mathcal{G}_\emptyset, B_n, D_n$ są najczęściej początkowymi grafami w wielu algorytmach przekształcających grafy.

8.2.2. Wycieczki w grafie nieskierowanym

8.2.2.1. Spójność grafu.

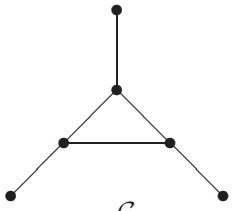
Definicja 8.2.1. *Jeśli dla każdych dwóch wierzchołków $u, v \in V(\mathcal{G})$ istnieje droga łącząca u i v to taki graf nazywamy spójnym.* □

Definicja 8.2.2. *Jeśli istnieje droga łącząca wierzchołek v z wierzchołkiem w to mówimy, że wierzchołek w jest osiągalny z wierzchołka v. W grafie spójnym każde dwa wierzchołki są osiągalne.* □

Usunięcie w grafie $(V(\mathcal{G}), E(\mathcal{G}))$ wierzchołka $v \in V(\mathcal{G})$ polega na tym, że tworzymy graf H z wierzchołkami $V(\mathcal{H}) = V(\mathcal{G})\backslash\{v\}$ oraz z krawędziami $E(\mathcal{H}) = \{\{u, w\} \in E(\mathcal{G}) : u \neq v \wedge w \neq v\}$. Usunięcie w grafie $(V(\mathcal{G}), E(\mathcal{G}))$ krawędzi $e \in E(\mathcal{G})$ oznacza utworzenie grafu H z $V(\mathcal{H}) = V(\mathcal{G})$ oraz $E\mathcal{H}) = E(\mathcal{G})\backslash\{e\}$. Rozspójnienie grafu oznacza usunięcie krawędzi lub wierzchołków lub i krawędzi i wierzchołków tak, aby powstał graf niespójny.

Problem 8.2.1. *Jak rozspójnić graf?* □

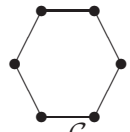
Definicja 8.2.3. *Najmniejszą liczbę usuniętych wierzchołków powodujących rozspójnienie grafu nazywamy spójnością wierzchołkową grafu i oznaczamy $\kappa(\mathcal{G})$. Najmniejszą liczbę usuniętych krawędzi powodujących rozspójnienie grafu nazywamy spójnością krawędziową grafu $\lambda(\mathcal{G})$.* □



$\mathcal{G}_1$

$\kappa(\mathcal{G}_1) = 1$

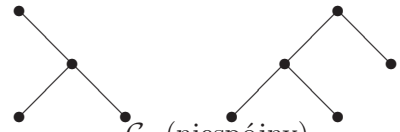
$\lambda(\mathcal{G}_1) = 1$



$\mathcal{G}_2$

$\kappa(\mathcal{G}_2) = 3$

$\lambda(\mathcal{G}_2) = 3$

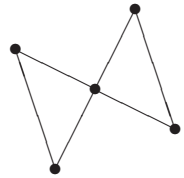


$\mathcal{G}_3$ (niespójny)

$\kappa(\mathcal{G}_3) = 0$

$\lambda(\mathcal{G}_3) = 0$

Uwaga!!! Te liczby niekoniecznie są sobie równe. Np.



$\kappa(\mathcal{G}_4) = 1$
$\lambda(\mathcal{G}_4) = 2$
$\mathcal{G}_4$

Problem 8.2.2. *Jak sprawdzić czy graf jest spójny? Zastanów się, co będzie wskazywać macierz sąsiedztwa przemnożona przez siebie, tzn* $M_{\mathcal{G}}^2 = M_{\mathcal{G}} \times M_{\mathcal{G}}$ *a co będą wskazywać kolejne macierze* $M_{\mathcal{G}}^k, k = 3, 4, ...$ □

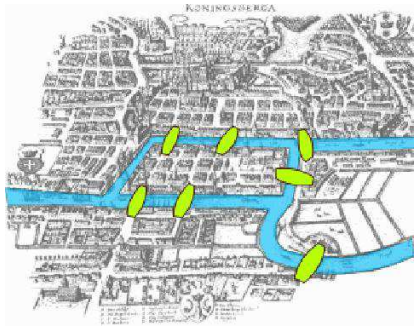
Ćwiczenie 8.2.1. *Znajdź spójność wierzchołkową i krawędziową grafów:* $W_n, C_n, SC_n, K_n, K_{r_1,r_2}$ *i Pet dla różnych wartości parametrów* $r_1, r_2, n \in \mathcal{N}$. *Jaka jest spójność wierzchołkowa i krawędziowa grafów dwudzielnych z rysunku 8.8?* □

8.2.2.2. Droga i cykl Eulera

Leonhard Euler stanął przed następującym problemem:

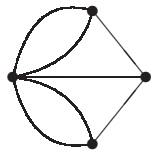
W Królewcu (wówczas Königsbergu) na rzece Pregole, na której są dwie wyspy wybudowano siedem mostów łączące wyspy ze sobą, oraz z oboma brzegami rzeki. Układ mostów został przedstawiony na rys. 8.13. Pytanie, jakie zostało postawione Eulerowi, to czy można tak ułożyć spacer po wszystkich mostach Królewca, by po każdym moście przejść tylko jeden raz i wrócić do punktu startowego?

Rysunek 8.13. Mapa mostów w Królewcu



Euler oczywiście odpowiedział na zadane mu pytanie. Powyższy problem można przedstawić w języku grafów. Niech każdy spójny kawałek lądu w Królewcu odpowiada wierzchołkowi. Otrzymamy w ten sposób dwa wierzchołki odpowiadające wyspom oraz dwa obu brzegom Pregoły. Most pomiędzy dwoma kawałkami lądu będziemy interpretować jako krawędź łączącą wierzchołki odpowiadające tym skrawkom lądu. W ten sposób otrzymamy graf (nie będący grafem prostym) jak na rys. 8.14.

Rysunek 8.14. Graf mostów Królewca



Definicja 8.2.4. *Mówimy, że graf $\mathcal{G}$ jest grafem Eulera (jest eulerowski) lub że ma cykl Eulera jeżeli istnieje cykl przechodzący przez wszystkie krawędzie grafu (oczywiście przez każdą tylko jeden raz).*

Niech $u \in V(\mathcal{G})$ będzie dowolnym wierzchołkiem grafu $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$. Przez $deg_{\mathcal{G}}(u)$ oznaczać będziemy liczbę krawędzi zawierających wierzchołek u. □

Twierdzenie 8.2.1. (Euler)

$$\sum_{u \in V(\mathcal{G})} deg_{\mathcal{G}}(u) = 2\overline{\overline{E(\mathcal{G})}}.$$ □

Problem 8.2.3. *Jakie wartości liczbowe mogą przyjmować stopnie wierzchołków grafu prostego?* □

Definicja 8.2.5. *Ciąg $\{d_1, d_2, ..., d_n\}$ nazywamy ciągiem graficznym jeżeli istnieje prosty graf o n wierzchołkach (bez pętli i krawędzi wielokrotnych) taki, że liczby te są stopniami wierszchołków tego grafu.* □

Twierdzenie 8.2.2. (P. Erdös, T. Gallai 1960) *Nierosnący ciąg liczb naturalnych* $\{d_1, .., d_n\}$ *jest ciągiem graficznym wtedy i tylko wtedy, gdy* $\sum_{i=1}^{n} d_i$ *jest liczbą parzystą oraz dla* $k = 1, 2, ..., n$,

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min\{k, d_i\}. \qquad \square$$

Ćwiczenie 8.2.2. *Czy ciąg* $\{4, 2, 4, 4, 3, 3, 4\}$ *jest graficzny? Jeśli tak, to narysuj graf prosty, którego stopniami wierzchołków są te liczby. Czy ten graf ma drogę/cykl Eulera? Jeśli ma to znajdź je w narsowanym grafie. Czy ciąg* $\{6, 3, 1, 1, 1\}$ *jest graficzny? Jeśli nie to czy istnieje graf, niekoniecznie prosty, (z pętlami i krawędziami wielokrotnymi) którego stopnie wierzchołków tworzą ten ciąg? Jeśli tak to narysuj ten graf. Czy istnieje dowolny graf którego wierzchołki tworzą ciąg* $\{2, 3, 1, 1, 2\}$*? Odpowiedź uzasadnij.* $\square$

Definicja 8.2.6. *Graf* $\mathcal{G}$ *nazywamy* r*-regularnym, jeśli*

$$\forall_{u \in V(\mathcal{G})} deg_{\mathcal{G}}(u) = r. \qquad \square$$

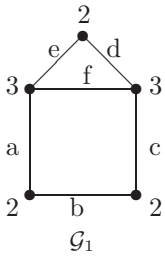
Uwaga 8.2.3. *Niech* $n = \overline{\overline{V(\mathcal{G})}}$*. Graf* $n-1$*-regularny o* n *wierzchołkach jest grafem pełnym* K_n*.* $\square$

Twierdzenie 8.2.4. *Liczba wierzchołków nieparzystego stopnia jest parzysta.* $\square$

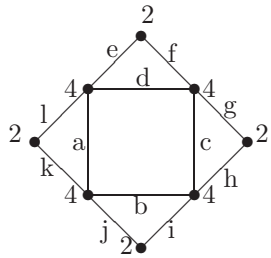
Twierdzenie 8.2.5. (Euler) *Graf skończony i spójny, w którym każdy wierzchołek ma stopień parzysty ma cykl Eulera.* $\square$

Twierdzenie 8.2.6. *Graf skończony i spójny, który ma dokładnie dwa wierzchołki stopnia nieparzystego ma drogę Eulera.* $\square$

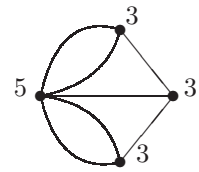
Rysunek 8.15. Droga i cykl Eulera



$\mathcal{G}_1$
nie jest grafem Eulera
droga Eulera: *abcdef*



$\mathcal{G}_2$ jest grafem Eulera
cykl Eulera: *abcdefghijkl*



$\mathcal{G}_3$ nie jest grafem Eulera
nie ma drogi
ani cyklu Eulera

Na rysunku 8.15 każdy wierzchołek opisany jest jego stopniem. Przedstawiamy tu przykłady drogi i cyklu Eulera.

Algorytm 8.1. Algorytm Fleury'ego znalezienia drogi (cyklu) Eulera

Krok 1: Wybierz dowolny wierzchołek v nieparzystego stopnia, jeśli taki istnieje, a jeśli nie, to wybierz dowolny wierzchołek v. Niech $VS = \{v\}$, $ES = \emptyset$.
Krok 2: Jeśli z wierzchołka v nie wychodzi żadna krawędź to zatrzymaj się.
Krok 3: Jeśli pozostała jedna krawędź wychodząca z wierzchołka v powiedzmy $e = \{v, w\}$ to usuń e z $E(\mathcal{G})$ oraz v z $V(\mathcal{G})$ i przejdź do kroku 5.
Krok 4: Jeżeli pozostała więcej niż jedna krawędź wychodząca z v to wybierz taką krawędź powiedzmy $e = \{v, w\}$ po której usunięciu graf zostanie spójny i usuń ją z $E(\mathcal{G})$.
Krok 5: Dołącz w na końcu ciągu VS i dołącze na końcu ciągu ES. Zastąp v wierzchołkiem w i przejdź do kroku 2.

8.2.2.3. Droga i cykl Hamiltona

Inny ciekawy problem można przedstawić na przykładzie firmy rozwożącej przesyłki. Dotyczy on pracy kuriera mającego rozwieść przesyłki do odbiorców w ten sposób, by odwiedzić każdego klienta jedynie raz, a na końcu wrócić do siedziby firmy. Każda z dróg (krawędzi grafu) ma swoją wagę (długość drogi w kilometrach). Jak znaleźć najkrótszą taką drogę? Jest to tzw. **problem komiwojażera.**

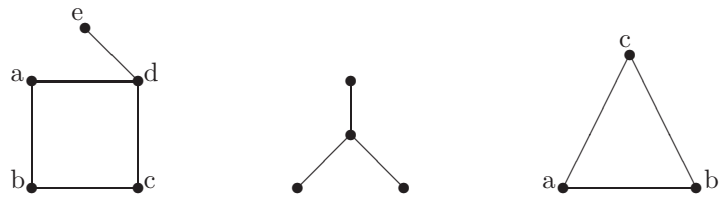
Rysunek 8.16. Problemy niemieckiego komiwojażera

Definicja 8.2.7. *Cykl Hamiltona to cykl przechodzący przez wszystkie wierzchołki grafu (czyli ścieżka zamknięta odwiedzająca każdy wierzchołek dokładnie raz). Graf hamiltonowski to graf posiadający cykl Hamiltona. Droga Hamiltona to ścieżka przechodząca przez wszystkie wierzchołki, każdy odwiedzając jedynie jeden raz.* □

W odróżnieniu od grafów eulerowskich, grafy hamiltonowskie nie posiadają prostej i szybkiej w użyciu charakteryzacji. Nie znana jest żadna

metoda, pozwalająca szybko stwierdzić czy dany graf jest hamiltonowski. Są natomiast znane pewne warunki wystarczające na to, by graf był hamiltonowski.

Rysunek 8.17. Droga i cykl Hamiltona



$\mathcal{H}_1$ bez cyklu Hamiltona ale z drogą Hamiltona: *abcde*

$\mathcal{H}_2$ nie ma drogi Hamiltona

$\mathcal{H}_3$ graf Hamiltona cykl Hamiltona: *abc*

Twierdzenie 8.2.7. (Dirac) *Graf prosty $\mathcal{G}$ taki, że*

(i) $\overline{\overline{V(\mathcal{G})}} = n \geq 3$

(ii) $\forall_{v \in V(\mathcal{G})} deg_{\mathcal{G}}(v) \geq \frac{n}{2}$,

jest grafem hamiltonowskim.

Dowód. Jeśli graf spełnia (ii) to dla każdych dwóch wierzchołków u i v zachodzi $deg_{\mathcal{G}}(u) + deg_{\mathcal{G}}(v) \geq n$ a więc to twierdzenie wynika z twierdzenia 8.2.8 (Ore). □

Twierdzenie 8.2.8. (Ore) *Graf prosty $\mathcal{G}$ taki, że*

(i) $\overline{\overline{V(\mathcal{G})}} = n \geq 3$

(ii) $\forall_{\{u,v\} \notin E(\mathcal{G})} deg_{\mathcal{G}}(u) + deg_{\mathcal{G}}(v) \geq n$,

jest grafem hamiltonowskim.

Dowód. (Nie wprost). Przypuśćmy, że twierdzenie jest fałszywe, czyli dla pewnej liczby n istnieje kontrprzykład $\mathcal{G}$ - graf, który spełnia założenie twierdzenia, ale nie jest Hamiltonowski. Spośród wszystkich takich grafów rozpatrzmy ten, który ma najmniesząą liczbę wierzchołków, a spośrod nich taki, dla którego wartość $\overline{\overline{E(\mathcal{G})}}$ jest maksymalna. Jest to podgraf pełnego grafu hamiltonowskiego K_n. Dodanie do $\mathcal{G}$ krawędzi z grafu K_n daje w wyniku graf, który nadal spełnia założenia twierdzenie i który ma więcej niż $\overline{\overline{E(\mathcal{G})}}$ krawędzi, a więc ze względu na wybór grafu $\mathcal{G}$ tak powstały graf będzie miał cykl Hamiltona. To znaczy, że $\mathcal{G}$ musi mieć (przynajmniej) drogę Hamiltona, określoną przez pewien ciąg wierzchołków $u_1, u_2, ..., u_n$. Ponieważ $\mathcal{G}$ nie ma cyklu Hamiltona, to nie istnieje krawędź łącząca u_n, u_1. Z kolei z założenia wiemy, że: $deg_{\mathcal{G}}(u_1) + deg_{\mathcal{G}}(u_n) \geq n$. Można teraz zdefiniować podzbiory A_1 i A_2 zbioru $\{2, 3, 4, ..., n\}$ takie, $A_1 = \{i : \{u_1, u_i\} \in E(\mathcal{G})\}$ oraz $A_2 = \{i : \{u_{i-1}, u_n\} \in E(\mathcal{G})\}$ gdzie numeracja i jest taka, jak w drodze

Hamiltona. Mamy więc

$$\overline{\overline{A_1}} = deg_{\mathcal{G}}(u_1),\ \overline{\overline{A_2}} = deg_{\mathcal{G}}(u_n),\ \overline{\overline{A_1}} + \overline{\overline{A_2}} \geq n,\ \overline{\overline{A_1 \cup A_2}} \leq n-1,$$

zatem $A_1 \cap A_2 \neq \emptyset$ a więc istnieje $i \in A_1 \cap A_2$ wtedy $u_1, u_2, ...u_{i-1}u_n u_{n-1}$ $u_{n-2}...u_i u_1$ jest cyklem Hamiltona, co prowadzi do sprzeczności. □

Twierdzenie 8.2.9. *Jeśli $\mathcal{G}$ jest grafem prostym takim, że*
(i) $\overline{\overline{V(\mathcal{G})}} = n$,
(ii) $\overline{\overline{E(\mathcal{G})}} \geq \binom{n-1}{2} + 2$,
wtedy $\mathcal{G}$ jest grafem hamiltonowskim.

Dowód. Dla dowolnego prostego grafu $\mathcal{G}$ załóżmy (ii) i weźmy dowolne dwa wierzchołki u i v takie, że $\{u, v\} \notin E(\mathcal{G})$. Niech H będzie grafem G z którego usunięto wierzchołki v i u oraz zawierające je krawędzie. Ponieważ $\{u, v\} \notin E(\mathcal{G})$, więc usunęliśmy $deg_{\mathcal{G}}(v) + \deg_{\mathcal{G}}(u)$ krawędzi i dwa wierzchołki. H jest podgrafem K_{n-2} a więc

$$\binom{n-2}{2} = \overline{\overline{K_{n-2}}} \geq \overline{\overline{H}} \geq \binom{n-1}{2} + 2 - deg_{\mathcal{G}}(v) - \deg_{\mathcal{G}}(u),$$

a stąd

$$deg_{\mathcal{G}}(v) + \deg_{\mathcal{G}}(u) \geq \binom{n-1}{2} - \binom{n-2}{2} + 2 = n,$$

a więc $\mathcal{G}$ spełnia założenia twierdzenia 8.2.8. □

Twierdzenie 8.2.10. (Bondy'ego-Chvátala) *Jeśli $\mathcal{G}$ jest grafem o n wierzchołkach a $C(\mathcal{G})$ jego nadgrafem zbudowanym według reguły:*

> *Jeśli $\{u, v\} \notin E(\mathcal{G})$ oraz $deg_{\mathcal{G}}(v) + \deg_{\mathcal{G}}(u) \geq n$ to krawędź $\{u, v\}$ dodajemy do $C(\mathcal{G})$,*

to wtedy graf $\mathcal{G}$ jest hamiltonowski wtedy i tylko wtedy gdy $C(\mathcal{G})$ jest hamiltonowski. □

Twierdzenie 8.2.11. Droga i cykl Hamiltona dla grafów dwudzielnych

(i) Jeśli $\mathcal{G}(V_1, V_2)$ ma cykl Hamiltona to $\overline{\overline{V_1}} = \overline{\overline{V_2}}$.
(ii) Jeśli $\mathcal{G}(V_1, V_2)$ ma drogę Hamiltona to $|\overline{\overline{V_1}} - \overline{\overline{V_2}}| \leq 1$.
(iii) Jeśli $r_1 = r_2$ to K_{r_1,r_2} ma cykl Hamiltona.
(iv) Jeśli $|r_1 - r_2| \leq 1$ to K_{r_1,r_2} ma drogę Hamiltona. □

Prosty dowód wynika z przechodzenia raz z wierzchołka z V_1 do wierzchołka z V_2 potem znowu z V_1 itd.

Algorytm 8.2. Algorytm Robertsa-Floresa

Krok 1: Budujemy macierz następników: kolumny macierzy odpowiadają wierzchołkom i zawierają ich następniki w pewnej, na początku ustalonej, kolejności $S = \emptyset$

Krok 2: Rozpoczynamy z dowolnego wierzchołka v. Bierzemy pierwszy "dostępny" (który jeszcze nie został włączony do zbioru S wierzchołków budowanego cyklu) następnik z kolumny odpowiadającej v. Załóżmy, że jest to wierzchołek u. $S := S \cup \{u\}$. Następnie bierzemy pierwszy dostępny następnik wierzchołka u z kolumny odpowiadającej u, itd.

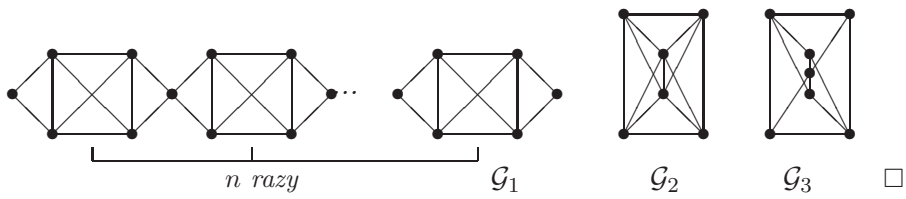
Krok 3: Mamy następujące możliwości:

1. Nie ma dostępnego następnika następuje krok powrotu wyrzucamy z S ostatnio dodany wierzchołek, wracamy do kolumny, z której został on wybrany i bierzemy kolejny dostępny następnik; następnie bierzemy pierwszy dostępny jego następnik, itd. (oczywiście za każdym razem, gdy nie ma dostępnego następnika następuje krok powrotu);

2. Zbiór S ma już moc n czyli znaleźliśmy już ścieżkę Hamiltona H z v do w, gdzie w jest ostatnio dodanym do S wierzchołkiem. Sprawdzamy, czy istnieje krawędź z w do v: jeśli TAK, to zapisujemy cykl Hamiltona $H + \{v, w\}$ i robimy krok powrotu gdy chcemy znaleźć wszystkie cykle Hamiltona lub STOP jeśli chcemy znaleźć tylko jeden cykl Hamiltona); jeśli NIE (nie istnieje krawędź z w do v) to krok powrotu.

Krok 4: Koniec następuje, gdy powrócimy do wierzchołka v i nie ma już dostępnych jego następników.

Problem 8.2.4. *Czy następujące grafy mają drogę (cykl) Eulera (Hamiltona)?*



□

8.2.3. Odwzorowania grafów

Definicja 8.2.8. Izomorfizm grafów. *Niech $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ oraz $\mathcal{F} = (V(\mathcal{F}), E(\mathcal{F}))$ będą dwoma grafami. Jeżeli istnieje bijekcja (funkcja różnowartościowa i "na") $f : V(\mathcal{G}) \longrightarrow V(\mathcal{F})$, taka, że*

$$\Big(< \{f(u), f(v)\}, k > \in E(\mathcal{F})\Big) \Longleftrightarrow \Big(< \{u, v\}, k > \in E(\mathcal{G})\Big)$$

(dla grafów skierowanych $\Big(< f(u), f(v), k > \in E(\mathcal{F})\Big) \Longleftrightarrow \Big(< u, v, k > \in E(\mathcal{G})\Big)\Big)$ to powiemy że grafy te są izomorficzne:

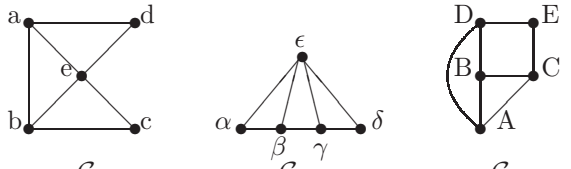
$$\mathcal{F} \cong \mathcal{G}.$$

□

Bezpośrednio z definicji wynika, że jeżeli f jest izomorfizmem grafów $\mathcal{F}$ i $\mathcal{G}$ to dla każdego wierzchołka $u \in V(\mathcal{F})$ jest $deg_{\mathcal{F}}(u) = deg_{\mathcal{G}}(f(u))$. Własność ta pomaga znajdować izomorfizm grafów.

Przykład 8.2.1. *Rozważmy trzy grafy:*

Rysunek 8.18. Izomorfizm grafów



wtedy

$\mathcal{G}_1$		$\mathcal{G}_2$		$\mathcal{G}_3$	
wierzchołek	*stopień*	*wierzchołek*	*stopień*	*wierzchołek*	*stopień*
a	3	α	2	A	3
b	3	β	3	B	3
c	2	γ	3	C	3
d	2	δ	2	D	3
e	4	ϵ	4	E	2

a więc w grafach $\mathcal{G}_1$ i $\mathcal{G}_2$ są dwa wierzchołki stopnia 2, dwa wierzchołki stopnia 3 i jeden stopnia 4, podczas gdy w grafie $\mathcal{G}_3$ jest jeden wierzchołek stopnia 2 i cztery wierzchołki stopnia 3. Na pewno więc $\mathcal{G}_1 \ncong \mathcal{G}_3$ oraz $\mathcal{G}_2 \ncong \mathcal{G}_3$. Musimy jeszcze rozstrzygnąć, czy $\mathcal{G}_1 \cong \mathcal{G}_2$? Gdyby f było takim izomorfizmem to $f(e) = \epsilon$ (bo jest jeden wierzchołek stopnia 4). Dalej możemy założyć, że albo $f(d) = \alpha$ albo $f(d) = \delta$. Przyjmijmy $f(d) = \alpha$, wtedy ponieważ d jest połączone krawędzią z a a α krawędzią z β otrzymujemy $f(a) = \beta$. Rozumując analogicznie dalej otrzymujemy

$$f : \begin{cases} a & \longrightarrow & \beta, \\ b & \longrightarrow & \gamma, \\ c & \longrightarrow & \delta, \\ d & \longrightarrow & \alpha, \\ e & \longrightarrow & \epsilon, \end{cases}$$

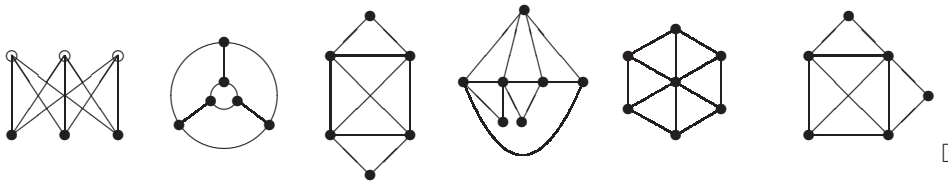
a więc $\mathcal{G}_1 \cong \mathcal{G}_2$. Rozważając $f(d) = \delta$ otrzymujemy inny izomorfizm grafów $\mathcal{G}_1$ oraz $\mathcal{G}_2$:

$$f : \begin{cases} a & \longrightarrow & \gamma, \\ b & \longrightarrow & \beta, \\ c & \longrightarrow & \alpha, \\ d & \longrightarrow & \delta, \\ e & \longrightarrow & \epsilon. \end{cases}$$

Innych izomorfizmów tych grafów już nie ma. □

Problem 8.2.5. *Które z podanych poniżej grafów są izomorficzne:*

Rysunek 8.19. Izomorfizm grafów



□

Algorytm 8.3. **Prosty algorytm testowania izomorfizmu**

Dane: Grafy proste $\mathcal{G}$ i $\mathcal{F}$

Wynik: `izom` wskazuje czy są czy nie izomorficzne

```
izom=False
if  V(G)‾‾ ≠ V(F)‾‾  then  stop
if  E(G)‾‾ ≠ E(F)‾‾  then  stop
for  każdej bijekcji  f : V_G → V_F
     izom1=True
     for  każdej pary  u, v ∈ V_G
          if  u, v  są połączone krawędzią a  f(u), f(v)  nie lub
          u, v  nie są połączone krawędzią a  f(u), f(v)  jest to
          izom1=False;  exit  for
     if  izom1  then  izom=True;  stop
stop
```

8.2.4. Kolorowanie grafów

8.2.4.1. Grafy planarne

Definicja 8.2.9. *Graf planarny jest to graf, który można narysować na płaszczyźnie bez przecięć. Ścianą wewnętrzną grafu narysowanego na płaszczyźnie nazywamy część płaszczyzny wyznaczoną przez krawędzie tego grafu a nieograniczony obszar poza grafem to ściana zewnętrzna.* □

Na przykład graf $\mathcal{G}_1$ z Rysunku 8.18 ma trzy ściany wewnętrzne wyznaczone przez trójkąty; $\triangle(aed)$, $\triangle(aeb)$, $\triangle(ebc)$ oraz jedną ścianę zewnętrzną. Każdy graf ma dokładnie jedną ścianę zewnętrzną.

Twierdzenie 8.2.12. (Euler) *Jeżeli w grafie planarnym n jest liczbą wierzchołków, e liczbą krawędzi, a f liczbą ścian, to zachodzi równość*

$$n - e + f = 2.$$

Dowód. Indukcja. Każdy graf możemy uzyskać wychodząc z pojedynczego wierzchołka ($n = 1, e = 0, f = 1$) i dodając kolejne krawędzie. Dodawana krawędź może być dodana wraz z nowym wierzchołkiem (n i e zwiększają się o 1, f pozostaje bez zmian) lub może łączyć dwa wcześniej dołączone wierzchołki (e i f zwiększają się o 1,n nie zmienia się). □

Wniosek 8.2.1. *Grafy K_5 i $K_{3,3}$ nie są planarne.*

Dowód. Załóżmy, że graf K_5 jest planarny. Mamy $n = 5, e = 10, f = 2 - n + e = 7$. Ale każda ściana ma co najmniej 3 krawędzie i dlatego $3f \leq 2e$, a w rozważamym grafie $3 \cdot 7 = 21 > 2 \cdot 10 = 20$ i mamy sprzeczność. Przypuśćmy, że graf $K_{3,3}$ jest planarny. Wtedy $e = 9, n = 6, f = 2 + 9 - 6 = 5$. W grafie $K_{3,3}$ nie ma trójkątów i dlatego $4f \leq 2e$, a w rozważanym grafie $4 \cdot 5 = 20 > 2 \cdot 9 = 18$ i mamy sprzeczność. □

Twierdzenie 8.2.13. *Rozważamy graf prosty planarny. Wtedy:*

- *W grafie istnieje wierzchołek stopnia nie większego niż 5.*
- *Graf ma co najwyżej $3n - 6$ krawędzi.*

Dowód. Każda ściana ma co najmniej 3 krawędzie. Niech $f_3, f_4, f_5, ...$ oznaczają liczbę ścian o $3, 4, 5, ...$ krawędziach. Mamy

$$\begin{aligned} f &= f_3 + f_4 + f_5 + ... \\ 2e &= 3f_3 + 4f_4 + 5f_5 + ... \end{aligned}$$

Stąd $2e \geq 3f$ i dalej $3n - 6 = 3e - 3f \geq e$. Załóżmy, że stopień każdego wierzchołka jest nie mniejszy niż 6. Wtedy

$$n = n_6 + n_7 + n_8 + ...$$

gdzie $n_6, n_7, n_8, ...$ oznaczają liczby wierzchołków stopnia $6, 7, 8, ...$ odpowiednio oraz

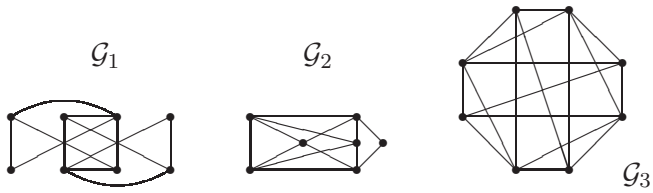
$$2e = 6n_6 + 7n_7 + 8n_8 + ...$$

Stąd

$$-12 = 6(-n + e - f) = (2e - 6n) + 2(2e - 3f) \geq 0$$

i mamy sprzeczność. □

Ćwiczenie 8.2.3. *Zbadaj planarność następujących grafów:*



□

Twierdzenie 8.2.14. (Kuratowskiego) *Graf jest planarny wtedy i tylko wtedy gdy nie zawiera podgrafu homeomorficznego z K_5 lub $K_{3,3}$ (tzn. takiego grafu, który różni się od wymienionych tym, że może mieć dodatkowe węzły na krawędziach).* □

Twierdzenie 8.2.15. *Graf jest planarny wtedy i tylko wtedy, gdy nie zawiera podgrafu ściągalnego do grafu K_5 lub $K_{3,3}$. Ściąganie grafu polega na kolejnym ściąganiu krawędzi. Natomiast ściąganie krawędzi polega na utożsamianiu wierzchołków, które łączy dana krawędź i pomijaniu ewentualnych pętli.* □

Graf Petersena jest nieplanarny i ściągalny do grafu K_5.

Definicja 8.2.10. *Graf dualny do danego grafu planarnego tworzymy umieszczając na każdej ścianie punkt i łącząc punkty leżące na sąsiednich ścianach krawędziami przecinającymi dzielące krawędzie (jeżeli na dzielących krawędziach leżą wierzchołki, to uzyskujemy krawędzie wielokrotne).*

Dowód. Alternatywny dowód twierdzenia Eulera. Dla danego grafu planarnego mającego n wierzchołków, e krawędzi i f ścian tworzymy graf dualny. Drzewo spinające graf (zob. podrozdz. 8.4.2) grafu dualnego ma $f-1$ krawędzi. Krawędzie te przecinają pewne krawędzie wyjściowego grafu. Pozostałe krawędzie wyjściowego grafu tworzą drzewo spinające o $n-1$ krawędziach. Suma przeciętych i nieprzeciętych krawędzi daje wszystkie krawędzie $(f-1)+(n-1)=e$ czyli $n-e+f=2$. □

8.2.4.2. Kolorowanie grafów

Definicja 8.2.11. *Najmniejszą liczbę kolorów potrzebnych do pomalowania wierzchołków grafu prostego $\mathcal{G}$ tak, aby sąsiednie wierzchołki miały różne kolory nazywamy liczbą chromatyczną i oznaczamy symbolem $\chi_{\mathcal{G}}$. Indeksem chromatycznym $\overline{\chi}_{\mathcal{G}}$ nazywamy najmniejszą ilość kolorów potrzebnych do pomalowania krawędzi grafu $\mathcal{G}$ tak, aby żadne dwie krawędzie mające wspólny wierzchołek nie były tego samego koloru.* □

Twierdzenie 8.2.16. *Jeśli w grafie prostym stopnie wierzchołków nie przekraczają d, to wystarczy $d+1$ kolorów do pokolorowania wierzchołków.*

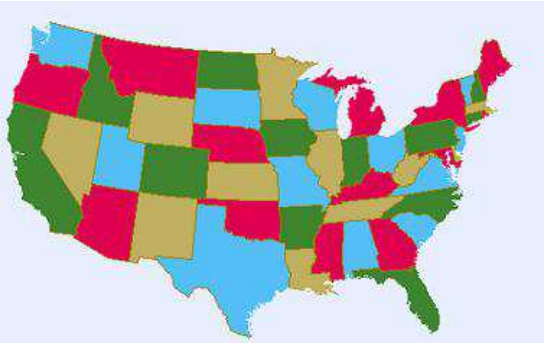
Dowód. Indukcja względem liczby wierzchołków. Usuwamy dowolny wierzchołek. Malujemy resztę za pomocą $d+1$ kolorów. Usunięty wierzchołek malujemy kolorem różnym od kolorów jego d sąsiadów. □

Twierdzenie 8.2.17. (Brooks) *Liczbę $d+1$ w Twierdzeniu 8.2.16 można zmiejszyć do d.* □

8.2.4.3. Kolorowanie grafów planarnych (kolorowanie map)

Mówimy, że dwa obszary (stany, województwa, państwa) na mapie mają wspólną granicę, jeżeli stykają się w więcej niż w pojedynczych punktach (!) tzn. jeżeli granica ich jest kawałkiem jakiejś krzywej, wykluczamy sytuację graniczenia tylko w jednym punkcie (jak to ma miejsce w USA w punkcie styku stanów: Nowy Meksyk, Arizona, Kolorado i Utah). Bez takiego założenia można konstruować mapy wymagające dowolnej liczby kolorów: wystarczy podzielić okrąg na części zawierające środek - do pomalowania użyjemy tylu kolorów, na ile części podzieliliśmy okrąg. Poza tym zakładamy, że jako obszary do pokolorowania rozważamy tylko zbiory spójne to znaczy obsazary te nie mogą si e dzielić na dwa rozłączne terytoria które muszą być pomalowane tą samą barwą (Jak stan Alaska na mapie USA). Poniżej przedstawiamy mapę stanów USA (bez Hawajów i Alaski) pomalowaną czterema barwami.

Rysunek 8.20. Problem 4 barw na przykładzie mapy USA



Krótka historia problemu 4 barw. W 1852 r. Francis Guthrie zauważył, że mapę hrabstw Anglii można pokolorować 4 barwami i zastanawiał się czy jest to prawdziwe dla każdej mapy, a jego brat Frederick spytał się o to Augusta De Morgana. Arthur Cayley w 1878 pierwszy opisał problem. W 1879 r. londyński prawnik A. B. Kempe opublikował "dowód" twierdzenia o 4 barwach, ale w 1890 Percy Heawood znalazł błąd w tym dowodzie. Hilbert przypuszczał (po pojawieniu się twierdzenia z logiki o tym że każda niesprzeczna logika jest nierozstrzygalna) że problem ten nie da się rozwiązać (rozstrzygnąć). Jednak poprawny dowód opracowali Kenneth Appel i Wolfgang Haken w 1977 r., ale dowód tym był poważnie wspierany programem komputerowym. W 1997 roku pojawił się nowy dowód tego wyniku, również wykorzystujący komputer, ale w sposób istotnie mniej skompliko-

wany od tego co robili Haken i Appel. Jego autorami są: Robertson, Sanders, Seymour i Thomas z Atlanty.

Problem kolorowania map można zamienić na opisany problem kolorowania wierzchołków grafu dualnego.

Twierdzenie 8.2.18. *Do pokolorowania grafu planarnego wystarczy* 6 *kolorów.*

Dowód. Indukcja względem liczby wierzchołków. Usuwamy wierzchołek stopnia 5 lub mniejszego (zob Twierdzenie 8.2.16 a). Kolorujemy resztę grafu, a potem kolorujemy usunięty wierzchołek kolorem różnym od kolorów 5 sąsiadów. □

Twierdzenie 8.2.19. *Do pokolorowania grafu planarnego wystarczy* 5 *kolorów.*

Dowód. Indukcja względem liczby wierzchołków. Jeśli wszystkie wierzchołki mają stopień mniejszy od 5, to malujemy bez trudności. W przeciwnym wypadku rozważamy wierzchołek v stopnia 5 połączony z wierzchołkami v_1, v_2, v_3, v_4, v_5. Dla pewnych i, j wierzchołki v_i, v_j nie są połączone. Inaczej nasz graf zawierałby graf K_5 i nie byłby planarny. Ściągamy krawędzie $\{v_i, v\}, \{v_j, v\}$. Tak otrzymany graf malujemy pięcioma kolorami. Następnie rozsuwamy ściągnięte krawędzie nadając wierzchołkom v_i, v_j kolor wierzchołka v, a wierzchołek v malujemy pozostałym piątym kolorem (teraz dwóch z pięciu sąsiadów v ma ten sam kolor). □

Twierdzenie 8.2.20. (Appel, Haken, komputer). *Do pokolorowania grafu planarnego wystarczą* 4 *kolory.* □

Ćwiczenie 8.2.4. *Spróbuj pokolorować mapę USA z rysunku 8.20 trzema kolorami. W którym miejscu pojawia się problem? Dlaczego? Czy potrafiłbyś opisać mały kawałek mapy, której nie da się pokolorować trzema barwami?* □

8.2.5. Kody Gray'a

Definicja 8.2.12. *Kodem Gray'a długości n nazywamy takie uporządkowanie wszystkich* 2^n *ciągów n cyfr binarnych, że kolejne ciągi różnią się dokładnie jedną cyfrą. To samo dotyczy pierwszego i ostatniego ciągu.* □

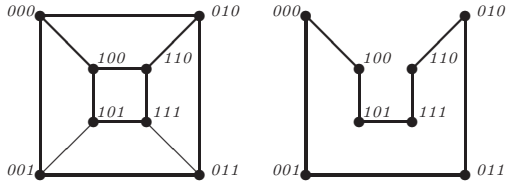
Na przykład dla $n = 2$ uporządkowanie $00, 01, 10, 11$ jest złe bo pomiędzy 01 a 10 jest różnica dwóch pozycji, również pomiędzy 11 (ostatnim) a 00 (pierwszym) jest różnica na dwóch pozycjach. Natomiast ciąg $01, 11, 10, 00$ jest dobrym ciągiem kodów Gray'a. Dla $n = 5$ ciąg kodów Gray'a podajemy w kolumnach poniżej:

Tabela 8.5. Kody Gray'a dla $n = 5$

00000	10001	10100	00101	01100	11101	11000	01001
00010	10011	10110	00111	01110	11111	11010	01011
00011	10010	10111	00110	01111	11110	11011	01010
00001	10000	10101	00100	01101	11100	11001	01000

Znajdowanie kodów Gray'a jest równoważne znajdowaniu cyklu Hamiltona w grafach Q_n (zob. podrozdz. 8.2.1) gdzie $V(\mathcal{G}) = \{0,1\}^n, E(\mathcal{G}) = \{uv : u, v \in V(\mathcal{G}) \wedge u \text{ i } v \text{ różnią się dokładnie 1 cyfrą}\}$.

Przykład 8.2.2. (n=3)



000
001
011
010
110
111
101
100

Q_3 *Cykl Hamiltona w* Q_3 *Kody Gray'a dla* $n = 3$ □

Kodów Gray'a można używać do etykietowania procesów w sieci będącej hiperkostką. Kody Gray'a mają wiele różnych zastosowań, na przykład do minimalizacji funkcji logicznych metodą tablicy Karnaugha.

Problem 8.2.6. *Czy mając dane kody Gray'a długości* $n-1$ *można z tych kodów otrzymać kody Gray'a długości* n*? Jak wygląda ta rekurencja? Czy jeśli każdy ciąg kodów Gray'a przekształcimy używając tej samej permutacji to czy też otrzymamy kody Gray'a? Czy jeśli kody Gray'a zapiszemy w odwrotnej kolejności to też otrzymamy kody Gray'a?* □

Problem 8.2.7. *Wykorzystując rozwiązanie problemu 8.2.6 napisz algorytm konstruujący kody Gray'a dowolnego rozmiaru i następnie permutujący te kody wg. zadanej permutacji.* □

Problem 8.2.8. *Jak wygląda konstrukcja ciągu kodów Gray'a dla alfabetu* {0,1,2} *długości 2? A długości 3? Czy też wtedy jest możliwa rekurencja taka jak w problemie 8.2.6?* □

8.2.6. Własności poznanych grafów

Własności wprowadzonych w podrozdziale 8.2.1 grafów streścimy w tabelce

Tabela 8.6. Własności podstawowych klas grafów

Graf	Eulera	Hamiltona	$\kappa(\mathcal{G})$	$\lambda(\mathcal{G})$	$\chi(\mathcal{G})$	$\overline{\chi}(\mathcal{G})$
SC_n	nie	nie	$1\,(n>2)$	$1\,(n>2)$	2	2
B_n	$n \geq 1$	$n \geq 1$	-	-	1	n
D_n	$2\vert n$	$n \geq 2$	-	-	2	n
K_n	$2 \not\vert n$	$n \geq 3$	$n-1$	$n-1$	n	$n-1\,(2\vert n, n \geq 2)$
						$n\,(2 \not\vert n, n \geq 2)$
$K_{m,n}$	$2\vert m$ i $2\vert n$	$m = n$	$m \wedge n$	$m \wedge n$	2	$m \vee n$
C_n	tak	$n \geq 1$	2	2	$2\,(2\vert n, n \geq 2)$	$2\,(2\vert n, n \geq 2)$
					$3\,(2 \not\vert n, n \geq 3)$	$3\,(2 \not\vert n, n \geq 3)$
W_n	nigdy	$n \geq 2$	3	3	$3\,(2\vert n, n \geq 4)$	n
					$4\,(2 \not\vert n, n \geq 3)$	
Q_n	$2\vert n$	$n \geq 2$	n	n	2^a	$?^b$
Pet	nie	nie	3	3	3	4

[a] Jeśli nie zawiera nieparzystego cyklu
[b] nieznane

gdzie $n \wedge m = \min\{n, m\}, n \vee m = \max\{n, m\}, 2\vert n$ gdy n jest parzyste i $2 \not\vert n$ gdy n nieparzyste.

8.3. Grafy skierowane

8.3.1. Wycieczki w grafie skierowanym

W grafie skierowanym dla każdego wierzchołka v można mówić o krawędziach wchodzących, to znaczy takich, których końcem jest wierzchołek v i krawędziach wychodzących, to znaczy takich których początkiem jest wierzchołek v. Ilość pierwszego typu krawędzi oznaczać będziemy przez $indeg_{\mathcal{G}}(v)$ a ilość drugiego typu krawędzi dla wierzchołka v oznaczać będziemy przez $outdeg_{\mathcal{G}}(v)$. Oczywiście dla każdego wierzchołka v mamy $deg_{\mathcal{G}}(v) = indeg_{\mathcal{G}}(v) + outdeg_{\mathcal{G}}(v)$.

Definicja 8.3.1. *Mówimy, że graf skierowany $\mathcal{G}$ jest grafem Eulera (jest eulerowski) lub że ma cykl Eulera jeżeli istnieje cykl skierowany (nie można przechodzić po krawędziach w przeciwnym niż oznaczony kierunek) przechodzący przez wszystkie krawędzie grafu (oczywiście przez każdą tylko jeden raz). Analogicznie mówimy że graf skierowany ma drogę Eulera.* □

Twierdzenie 8.3.1. *Skierowany spójny graf $\mathcal{G}$ ma drogę lecz nie cykl Eulera jeżeli istnieje dokładnie jeden wierzchołek v taki, że $outdeg_{\mathcal{G}}(v) = indeg_{\mathcal{G}}(v) + 1$, istnieje dokładnie jeden wierzchołek u taki, że $outdeg_{\mathcal{G}}(u) + 1 = indeg_{\mathcal{G}}(u)$ i dla wszystkich pozostałych wierzchołków w mamy $outdeg_{\mathcal{G}}(w) = indeg_{\mathcal{G}}(w)$. Wtedy wierzchołek v jest początkiem a u końcem tej drogi. Skierowany graf*

$\mathcal{G}$ ma cykl Eulera wtedy i tylko wtedy, gdy dla każdego wierzchołka v zachodzi $outdeg_{\mathcal{G}}(v) = indeg_{\mathcal{G}}(v)$. □

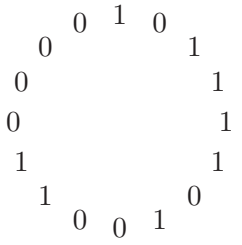
Dla grafów skierowanych konstrukcja drogi (cyklu) Eulera (Hamiltona) jest prostą modyfikacją algorytmów 8.1 i 8.2 odpowiednio.

Ćwiczenie 8.3.1. *Znajdź drogę (cykl) Eulera (Hamiltona) w grafie na rysunku 8.22.* □

8.3.2. Ciąg de Bruijna

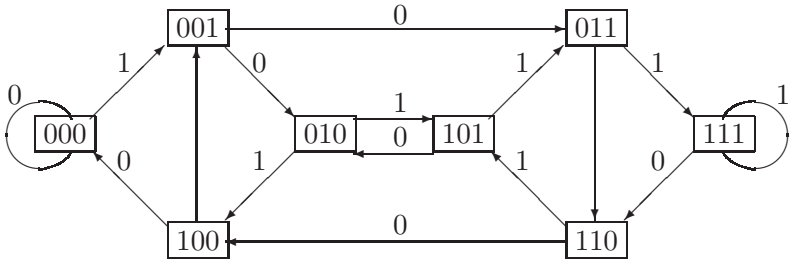
Definicja 8.3.2. *Ciągiem de Bruijna rzędu n nazywamy takie ustawienie* 2^n *cyfr* 0 *i* 1 *(powiedzmy* $c_1c_2, ..., c_{2^n}$*), że każdy ciąg długości n występuje w tym ustawieniu dokładnie raz, to znaczy ciągi:* $(c_1, .., c_n), (c_2, ..., c_{n+1})...(c_{2^n}c_1, ..., c_{n-1})$ *są różne.*

Rysunek 8.21. Ciąg de Bruijna dla $n = 4$



Aby wyjaśnić konstrukcję tego ciągu rozważmy następujący graf:

Rysunek 8.22. Graf de Brujina



Graf de Bruijna ma wierzchołki etykietowane ciągami długości 2^{n-1}. Ciąg de Bruijna jest konstruowany tak, że jeżeli mamy element np 0110 to następnym elementem musi być 110∗ gdzie w miejsce ∗ powinniśmy wstawić 0 lub 1. Oczywiście w całym ciągu musi być i 1100 i 1101. Graf de Bruijna skonstruowany jest tak, że połączone są ze sobą wierzchołki $c_0c_1c_2$ i c_1c_20 oraz c_1c_21. Wtedy problem znalezienia ciągu de Bruijna można interpreto-

wać jako znalezienie cyklu Eulera w skierowanym grafie de Bruijna. Jeśli rozważymy w grafie na rysunku 8.22 następujący cykl Eulera:

$$101 \xrightarrow{0} 011 \xrightarrow{1} 111 \xrightarrow{1} 111 \xrightarrow{0} 110 \xrightarrow{1} 101 \xrightarrow{0} 010 \xrightarrow{0} 100 \xrightarrow{1} 001$$
$$001 \xrightarrow{1} 011 \xrightarrow{0} 110 \xrightarrow{0} 100 \xrightarrow{0} 000 \xrightarrow{0} 000 \xrightarrow{1} 001 \xrightarrow{0} 010 \xrightarrow{1} 101$$

otrzymaliśmy ciąg de Bruijna z rysunku 8.21.

Problem 8.3.1. *Co należy wstawić w miejsce gwiazdek w ciągu de Brujina:* $110\star00100\star101011$*? Jak skonstruować ciąg de Bruijna długości* $n = 3$*? Czy istnieje uogólnienie ciągu de Bruijna dla alfabetu trzyznakowego* $\{0, 1, 2\}$*?* □

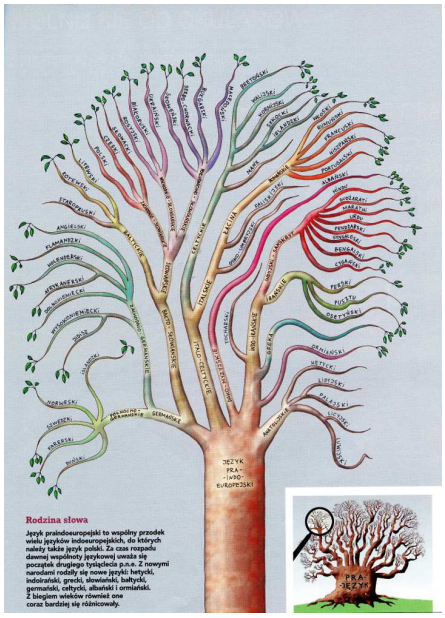
8.4. Drzewa

8.4.1. Własności drzew

Drzewo to prosty graf acykliczny i spójny. Nie ma pętli ani krawędzi wielokrotnych.

Pojęcie drzewa często występuje w praktyce. Na przykład możemy się spotkać z drzewami w genealogii, językoznawstwie czy filogenetyce.

Rysunek 8.23. Przykłady zastosowań drzew



Czasem definiuje się drzewa rekurencyjnie:

(i) Pojedynczy wierzchołek v jest drzewem
(ii) Jeżeli $T_1, ..., T_n$ są drzewami a v jest pojedynczym wierzchołkiem nie należącym do żadnego z tych drzew to graf T powstały przez połączenie wierzchołka v z dokładnie jednym wierzchołkiem z każdego z drzew T_i dla $1 \leq i \leq n$ jest również drzewem.

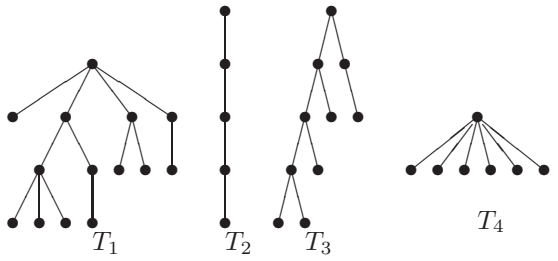
Jest dużo różnych charakteryzacji pojęcia drzewa:

Twierdzenie 8.4.1. *Niech $\mathcal{G}$ będzie grafem bez pętli i krawędzi wielokrotnych. Załóżmy, że $\overline{\overline{V(\mathcal{G})}} = n \geq 2$. Następujące warunki są równoważne:*
(i) $\mathcal{G}$ jest drzewem.
(ii) $\mathcal{G}$ jest acykliczny o $n-1$ wierzchołkach.
(iii) $\mathcal{G}$ jest spójny o $n-1$ krawędziach. □

Wierzchołki stopnia 1 nazywamy liśćmi.

Twierdzenie 8.4.2. *Niech $\mathcal{G}$ będzie grafem bez pętli i krawędzi wielokrotnych. Załóżmy, że $\overline{\overline{V(\mathcal{G})}} \geq 2$. Następujące warunki są równoważne:*
(i) $\mathcal{G}$ jest drzewem.
(ii) Dla każdych dwóch różnych wierzchołków grafu $\mathcal{G}$ istnieje dokładnie jedna droga łącząca te wierzchołki.
(iii) $\mathcal{G}$ jest spójny i usunięcie dowolnej krawędzi rozspójnia $\mathcal{G}$.
(iv) $\mathcal{G}$ jest acykliczny i dodanie dowolnej krawędzi powoduje powstanie cykli. □

Rysunek 8.24. Przykłady drzew



Z samej definicji drzewa T mamy $\kappa(T) = \lambda(T) = 1$. Ponadto żadne drzewo T o $\overline{\overline{V(T)}} \geq 2$ nie jest grafem Eulera ani Hamiltona, chociaż drzewo T_2 z rysunku 8.24 ma drogę Eulera i Hamiltona. Jeżeli w definicji drzewa pominiemy słowo "spójny" to tak powstały graf nazywamy lasem. Na przykład $T_1 \cup T_2 \cup T_3 \cup T_4$ z rysunku 8.24 to las złożony z czterech drzew.

8.4.2. Drzewo spinające graf

Definicja 8.4.1. *Niech $\mathcal{G} = (V(\mathcal{G}), E(\mathcal{G}))$ będzie dowolnym spójnym grafem. Drzewem $\mathcal{T} = (V(\mathcal{T}), E(\mathcal{T}))$ spinającym graf $\mathcal{G}$ nazywamy takie drze-*

wo, że

$$\begin{aligned} V(\mathcal{T}) &= V(\mathcal{G}), \\ E(\mathcal{T}) &\subseteq E(\mathcal{G}). \end{aligned}$$ □

Twierdzenie 8.4.3. *Każdy graf spójny ma drzewo spinające.* □

Definicja 8.4.2. *Niech będzie dany graf $\mathcal{G} = (V, E)$ a $\mathcal{H} \subseteq \mathcal{G}$ niech będzie podgrafem grafu $\mathcal{G}$. Funkcja f taka, że*

$$f : E(\mathcal{G}) \longrightarrow \mathcal{R}_+ \cup \{0\},$$

nazywamy wagą grafu $\mathcal{G}$. Wagę całego podgrafu definiuje się jako

$$\omega(\mathcal{H}) = \sum_{e \in E(\mathcal{H})} f(e).$$ □

Problem 8.4.1. *Jak dla dowolnego grafu $\mathcal{G}$ z wagą f znaleźć drzewo $\mathcal{T}$ spinające ten graf o minimalnej wadze $\omega(\mathcal{T})$?* □

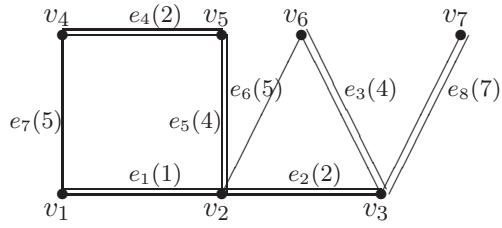
Algorytm 8.4. Algorytm Kruskala

Dane: Niech G będzie skończonym grafem spójnym z wagami, którego krawędzie uporządkowane są: $e_1 < e_2 < e_3 < ... < e_m$ tak, że $f(e_1) \leq f(e_2) \leq f(e_3) \leq ... \leq f(e_m)$.

Wynik: Zbiór E krawędzi minimalnego drzewa spinającego graf $\mathcal{G}$.

```
E:=∅
for j:=1 to m do
   {
     if E∪{e_j} jest acykliczny then
         E:=E∪{e_j}
    }
```

Rysunek 8.25. Algorytm Kruskala



Najpierw porządkujemy krawędzie $e_1 \leq e_2 \leq e_4 \leq e_3 \leq e_5 \leq e_7 \leq e_6 \leq e_8$ następnie wykonujemy kolejne kroki algorytmu

Tabela 8.7. Algorytm Kruskala

E	$E \cup \{e_j\}$	czy cykliczny
$\emptyset$	$\{e_1\}$	NIE
$\{e_1\}$	$\{e_1, e_2\}$	NIE
$\{e_1, e_2\}$	$\{e_1, e_2, e_4\}$	NIE
$\{e_1, e_2, e_4\}$	$\{e_1, e_2, e_4, e_3\}$	NIE
$\{e_1, e_2, e_4, e_3\}$	$\{e_1, e_2, e_4, e_3, e_5\}$	NIE
$\{e_1, e_2, e_4, e_3, e_5\}$	$\{e_1, e_2, e_4, e_3, e_5, e_7\}$	CYKL $\{e_1, e_5, e_4, e_7\}$
$\{e_1, e_2, e_4, e_3, e_5\}$	$\{e_1, e_2, e_4, e_3, e_5, e_6\}$	CYKL $\{e_2, e_3, e_6\}$
$\{e_1, e_2, e_4, e_3, e_5\}$	$\boxed{\{e_1, e_2, e_4, e_3, e_5, e_8\}}$	NIE

Jeśli graf $\mathcal{G}$ jest spójny to w wyniku otrzymujemy minimalne drzewo spinające graf $\mathcal{G}$. Jeśli graf $\mathcal{G}$ nie jest spójny to otrzymujemy las spinający graf $\mathcal{G}$.

Algorytm 8.5. Algorytm Prima

Dane: Niech G będzie skończonym grafem spójnym zwagami (krawędzie niekoniecznie uporządkowane)

Wynik: Zbiór E krawędzi minimalnego drzewa spinającego graf $\mathcal{G}$.

E:=$\emptyset$
{wybierz wierzcholek $v \in V(\mathcal{G})$}
V:=$\{v\}$
while $V \neq V(\mathcal{G})$ do
{
Wybierz krawędź $\{u, v\}$ ze zbioru $E(\mathcal{G})$o najmniejszej wadze tak, że $u \in V, v \in V(\mathcal{G}) - V$;
{Dopisz $\{u, v\}$ do E}
E:=E$\cup\{\{u, v\}\}$;
{Dopisz v do V}
V:=V$\cup\{v\}$;
}

Przykład działania algorytmu Prima dla grafu z rysunku 8.25.

Tabela 8.8. Algorytm Prima

E	V	$V(\mathcal{G})\backslash V$	krawędź	waga
$\emptyset$	$\{v_1\}$	$\{v_2, v_3, v_4, v_5, v_6, v_7\}$	$\boxed{\{v_1, v_2\} = e_1}$	1
			$\{v_1, v_4\} = e_7$	5
$\{e_1\}$	$\{v_1, v_2\}$	$\{v_3, v_4, v_5, v_6, v_7\}$	$\{v_1, v_4\} = e_7$	5
			$\boxed{\{v_2, v_3\} = e_2}$	2
			$\{v_2, v_5\} = e_5$	4
			$\{v_2, v_6\} = e_6$	5
$\{e_1, e_2\}$	$\{v_1, v_2, v_3\}$	$\{v_4, v_5, v_6, v_7\}$	$\{v_1, v_4\} = e_7$	5
			$\boxed{\{v_2, v_5\} = e_5}$	4
			$\{v_2, v_6\} = e_6$	5
			$\{v_3, v_6\} = e_3$	4
			$\{v_3, v_7\} = e_8$	7
$\{e_1, e_2, e_5\}$	$\{v_1, v_2, v_3, v_5\}$	$\{v_4, v_6, v_7\}$	$\{v_1, v_4\} = e_7$	5
			$\{v_2, v_6\} = e_6$	5
			$\{v_3, v_6\} = e_3$	4
			$\{v_3, v_7\} = e_8$	7
			$\boxed{\{v_4, v_5\} = e_4}$	2
$\{e_1, e_2, e_4, e_5\}$	$\{v_1, v_2, v_3, v_4, v_5\}$	$\{v_6, v_7\}$	$\{v_2, v_6\} = e_6$	5
			$\boxed{\{v_3, v_6\} = e_3}$	4
			$\{v_3, v_7\} = e_8$	7
$\{e_1, e_2, e_3, e_4, e_5\}$	$\{v_1, v_2, v_3, v_4, v_5, v_6\}$	$\{v_7\}$	$\boxed{\{v_3, v_7\} = e_8}$	7
$\{e_1, e_2, e_3, e_4, e_5, e_8\}$	$\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$	$\emptyset$		

Problem 8.4.2. *W Tabeli 8.9 podajemy odległości w milach pomiędzy miastami w USA które towarzystwo naftowe rurociągami przechodzącymi bezpośrednio pod tymi miastami. Ile co najmniej mil rur do tego potrzeba?*

Tabela 8.9. Tabela odległości miast

	Des Moines	Milwaukee	Minneapolis	Omaha	Pierre	Winnipeg
Bismarck	670	758	427	581	211	369
Des Moines		361	252	132	492	680
Milwaukee			332	493	690	759
Minneapolis				357	394	432
Omaha					391	650
Pierre						521

□

8.4.3. Algorytm Huffmana

W teorii komunikacji mamy często do czynienia z sytuacją gdy jest pewien zbiór komunikatów, powiedzmy $K = \{k_1, k_2, ..., k_n\}$ o prawdopodobieństwach pojawienia się (częstościach) $P = \{p_1, p_2, ..., p_n\}$, odpowiednio. Oczywiście, zakładać będziemy, że $0 \leq p_i \leq 1, 1 \leq i \leq n$ oraz $\sum_{i=1}^{n} p_i = 1$.

Załóżmy również, że mamy pewien alfabet $A = \{a_1, a_2, ..., a_l\}$ którego ciągami chcielibyśmy zakodować wszystkie komunikaty. Bez zmniejszenia ogólności rozważań możemy założyć, że alfabet $A = \{0, 1\}$ jest dwuznakowy. Ponieważ nie rozważamy specjalnego znaku - końca komunikatu, chcemy, żeby ciąg zer i jedynek na wyjściu dał się jednoznacznie "odczytać" komunikatami z K. Załóżmy, że mamy pewien sposób kodowania komunikatów z K ciągami 0 i 1 i niech dla tego sposobu kodowania $S = \{l_1, l_2, ..., l_n\}$ będą to długości kodów komunikatów $\{k_1, k_2, ..., k_n\}$ odpowiednio. Jesteśmy zainteresowani minimalizacją funkcji

$$L_S = \sum_{i=1}^{n} l_i p_i.$$

W teorii komunikacji dużą rolę odgrywa **entropia**

$$H = -\sum_{i=1}^{n} p_i \log_2(p_i).$$

Można udowodnić, że dla każdego poprawnego sposobu kodowania S wielkość

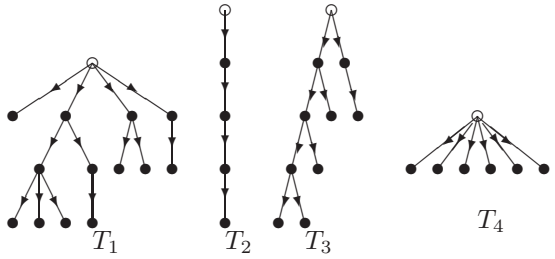
$$R_S = L_S - H,$$

jest nieujemna, a jeżeli przyjmuje wartość 0 to kod jest optymalny, są jednak układy prawdopodobieństw dla których nigdy nie przyjmie wartości 0. Wielkość R nazywamy **redundancją** (nadmiarowością) komunikatu. Redundancja współczesnego języka polskiego wynosi około 73%, to znaczy, że mniej więcej 4 razy moglibyśmy przeciętnie skrócić wypowiedź, bez utraty informacji. Redundancja w komunikacji międzyludzkiej jest jednak potrzebna do odtworzenia uszkodzonego (gdy na przykład czegoś nie dosłyszeliśmy) komunikatu. Również przy transmisjach komputerowych stosuje się redundancję.

Problemem, którym zajmować się będziemy w tym podrozdziale, jest problem konstrukcji optymalnych, o minimalnej redundancji, kodów spośród grupy tak zwanych **kodów migawkowych,** to znaczy kodów których żadne słowo kodowe nie jest przedrostkiem innego słowa kodowego. Takie kody można jednoznacznie opisać skierowanymi drzewami.

Jeżeli w drzewie nieskierowanym wyróżnimy jeden wierzchołek (rdzeń, korzeń drzewa) to takie drzewo staje się już drzewem skierowanym. Na przykład rozważając drzewa z rysunku 8.24 po wyborze korzenia otrzymujemy

Rysunek 8.26. Przykłady drzew skierowanych



Definicja 8.4.3. *Skierowane drzewo nazywamy drzewem z wagami jeżeli jest określona funkcja f przyporządkowująca liściom drzewa liczby rzeczywiste. Poziomem węzła v drzewa nazywamy długość drogi (ilość krawędzi) $l(v)$ jakie prowadzą od rdzenia drzewa do węzła v. Wagą drzewa nazywamy wielkość*

$$\sum_{v\in E(T), v \text{ jest liściem}} f(v)l(v). \qquad \square$$

Na przykład, wagi i poziomy liści z rysunku 8.26 występują w tabelce 8.10 w drugim i czwartym wierszu a waga całego drzewa została obliczona we wzorze (8.4.1) i wynosi 3.13.

Definicja 8.4.4. *Drzewo skierowane T nazywamy drzewem binanarnym jeśli dla każdego węzła v jest $outdeg_T(v) = 2, outdeg_T(v) = 1$ lub $outdeg_T(v) = 0$ (wtedy v jest liściem).* $\square$

Przykład 8.4.1. *Rozważmy cztery następujące sposoby kodowania komunikatów o prawdopodobieństwach podanych w tabelce*

Tabela 8.10. Przykłady kodów

Komunikat	Częstość	*Kod* 1		*Kod* 2		*Kod* 3		*Kod* 4	
		K_1	S_1	K_2	S_2	K_3	S_3	K_4	S_4
a	$\frac{1}{2}$	0	1	10	2	0	1	01	2
b	$\frac{1}{4}$	1	1	1000	4	10	2	000	3
c	$\frac{1}{8}$	00	2	1001	4	110	3	1	1
d	$\frac{1}{8}$	01	2	1011	4	111	3	001	3

Teraz

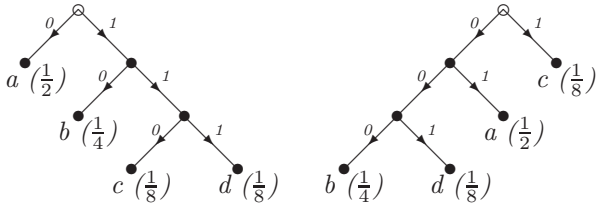
$$L_1 = \frac{1}{2}\cdot 1 + \frac{1}{4}\cdot 1 + \frac{1}{8}\cdot 2 + \frac{1}{8}\cdot 2 = 1\frac{1}{4} \text{ bitów},$$

$$
\begin{aligned}
L_2 &= \frac{1}{2}\cdot 2+\frac{1}{4}\cdot 4+\frac{1}{8}\cdot 4+\frac{1}{8}\cdot 4=3 \text{ } bity,\\
L_3 &= \frac{1}{2}\cdot 1+\frac{1}{4}\cdot 2+\frac{1}{8}\cdot 3+\frac{1}{8}\cdot 3=1\frac{3}{4} \text{ } bitów,\\
L_4 &= \frac{1}{2}\cdot 2+\frac{1}{4}\cdot 3+\frac{1}{8}\cdot 1+\frac{1}{8}\cdot 1=2 \text{ } bity,\\
H &= -\frac{1}{2}\log_2(\frac{1}{2})-\frac{1}{4}\log_2(\frac{1}{4})-\frac{1}{8}\log_2(\frac{1}{8})-\frac{1}{8}\log_2(\frac{1}{8})=1\frac{3}{4} \text{ } bitów.
\end{aligned}
$$

a stąd

$$
\begin{aligned}
R_1 &= L_1-H=-\tfrac{1}{2} \text{ } bita, \quad & R_3 &= L_3-H=0 \text{ } bitów,\\
R_2 &= L_2-H=1\tfrac{1}{4} \text{ } bitów, \quad & R_4 &= L_4-H=\tfrac{1}{4} \text{ } bitów.
\end{aligned}
$$

Zwróćmy uwagę, że kod 1 nie jest kodem jednoznacznie dekodowalnym. Na przykład ciąg na wyjściu 00101 *może być odczytany jako* aabab *lub jako* add*, jest to więc kod z punktu widzenia teorii kodowania błędny. Pozostałe kody są jednoznacznie dekodowalne (chociaż nie tak łatwo to sprawdzić). Kod 2 nie jest kodem migawkowym, bowiem słowo kodowe* 10 *(komunikat a) jest przedrostkiem pozostałych trzech kodów. Kody 3 i 4 są kodami migawkowymi i mogą być przedstawione za pomocą skierowanego drzewa binarnego:*



Kod 3 *Kod 4*

Kod 3, ponieważ ma redundancję równą 0 *bitów i jest jednoznacznie dekodowalny jest też kodem optymalnym. Powstaje problem jak dla zadanego ciągu prawdopodobieństw skonstruować optymalny kod migawkowy?* □

Algorytm Huffmana konstruuje dla zadanego ciągu liczb optymalne drzewo binarne to znaczy drzewo binarne o najmniejszej możliwej wadze.

Algorytm 8.6. Algorytm Huffmana

HUFFMAN(L)

Dane: Ciąg $L=\{\omega_1,\omega_2,...,\omega_t\}$

Wynik: Optymalne drzewo binarne $T(L)$ dla ciągu L

Jeżeli $t = 2$ to niech $T(L)$ będzie drzewem z dwoma liśćmi ω_1 i ω_2 w przeciwnym razie

1. Znajdź dwa najmniejsze wyrazy w ciągu L powiedzmy u i v
2. Niech L' będzie listą powstałą z L przez usunięcie u i v i wstawienie $u + v$.
3. Wykonaj algorytm HUFFMAN(L') by otrzymać drzewo $T(L')$
4. utwórz drzewo $T(L)$ z drzewa $T(L')$ zastępując liść wagi $u + v$ w drzewie $T(L')$ poddrzewem mającym dwa liście o wagach u i v

Przykład 8.4.2. *Danych jest dwanaście komunikatów o prawdopodobieństwach*

k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
$0,01$	$0,02$	$0,12$	$0,07$	$0,10$	$0,08$	$0,31$	$0,05$	$0,02$	$0,08$	$0,06$	$0,08$

Znajdźmy optymalny sposób kodowania tych komunikatów alfabetem dwuznakowym (np. alfabetem Morse'a traktując 1 *jak* $-$ *a* 0 *jak* $\cdot$*). Kolejne przekształcenia podajemy w tabeli (kolejne ciągi* L'*) a w nawiasach klamrowych pamiętamy liczby, których sumy dały odpowiednie wagi*

Rysunek 8.27. Śledzenie algorytmu Huffmana

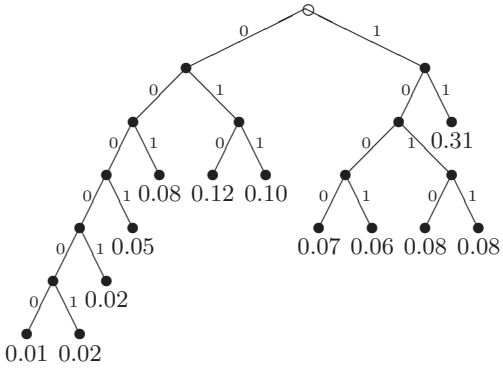
$\underline{0.01}$	$\underline{0.02}$	0.12	0.07	0.10	0.08	0.31	0.05	0.02	0.08	0.06	0.08
$\underline{0.03}\{0.01, 0.02\}$		0.12	0.07	0.10	0.08	0.31	0.05	$\underline{0.02}$	0.08	0.06	0.08
$\underline{0.05}\{\{0.01, 0.02\}, 0.02\}$			0.12	0.07	0.10	0.08	0.31	$\underline{0.05}$	0.08	0.06	0.08
$0.10\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}$				0.12	$\underline{0.07}$	0.10	0.08	0.31	0.08	$\underline{0.06}$	0.08
$0.10\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}$				0.12	$0.13\{0.07, 0.06\}$		0.10	$\underline{0.08}$	0.31	$\underline{0.08}$	0.08
$\underline{0.10}\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}$				0.12	$0.13\{0.07, 0.06\}$		0.10	$0.16\{0.08, 0.08\}$		0.31	$\underline{0.08}$
$0.18\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}$					$\underline{0.12}$	$0.13\{0.07, 0.06\}$		$\underline{0.10}$	$0.16\{0.08, 0.08\}$		0.31
$0.18\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}$					$0.22\{0.12, 0.10\}$		$\underline{0.13}\{0.07, 0.06\}$		$\underline{0.16}\{0.08, 0.08\}$		0.31
$\underline{0.18}\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}$					$\underline{0.22}\{0.12, 0.10\}$		$0.29\{\{0.07, 0.06\}, \{0.08, 0.08\}\}$				0.31
$0.40\{\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}, \{0.12, 0.10\}\}$							$\underline{0.29}\{\{0.07, 0.06\}, \{0.08, 0.08\}\}$				$\underline{0.31}$
$0.40\{\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}, \{0.12, 0.10\}\}$							$\underline{0.60}\{\{\{0.07, 0.06\}, \{0.08, 0.08\}\}, 0.31\}$				
$1.00\{\{\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}, \{0.12, 0.10\}\}, \{\{\{0.07, 0.06\}, \{0.08, 0.08\}\}, 0.31\}\}$											

Otrzymany ciąg

$$\{\{\{\{\{\{0.01, 0.02\}, 0.02\}, 0.05\}, 0.08\}, \{0.12, 0.10\}\}, \{\{\{0.07, 0.06\}, \{0.08, 0.08\}\}, 0.31\}\}$$

umożliwia nam skonstruowanie wymaganego drzewa:

Rysunek 8.28. Drzewo $T(L)$



a to prowadzi do kodu

Tabela 8.11. Wynik algorytmu Huffmana

k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}
$0,01$	$0,02$	$0,12$	$0,07$	$0,10$	$0,08$	$0,31$	$0,05$	$0,02$	$0,08$	$0,06$	$0,08$
000000	000001	010	1000	011	1010	11	0001	00001	1011	1001	001
6	6	3	4	3	4	2	4	5	4	4	3

dla którego średnia długość słowa kodowego wynosi

$$L = 0.01 \cdot 6 + 0.02 \cdot 6 + 0.12 \cdot 3 + ... + 0.08 \cdot 3 = 3.13 \tag{8.4.1}$$

podczas gdy

$$H = -0.01 \log_2(0.01) - 0.02 \log_2(0.02) - ... - 0.08 \log_2(0.08) = 3.117958...$$

a więc redundancja wynosi $R = 0,012042...$ *bita.* □

Ćwiczenie 8.4.1. *Narysuj* $T(L)$*, gdzie* $L = \{3.2; 4.1; 1.8; 1.1; 2.5; 1.1; 7.6; 5.5; 4.1; 3.3; 7.1; 8.1; 4.0; 2.5; 6.2; 7.7\}$. □

Problem 8.4.3. *Drzewo nazwiemy drzewem* r*-narnym jeżeli dla wszystkich węzłów* v *mamy* $outdeg(v) \leq r$*. Jak skonstruować algorytm Huffmana dla takich drzew? A jak konstruować dla drzew, które mają ten sam stopień krawędzi wychodzących na tym samym poziomie drzewa?* □

8.5. Zastosowania teorii grafów

Grafy i sieci występują w tak wielu dziedzinach życia, że nie sposób wymienić nawet najważniejszych zastosowań. My tutaj naszkicujemy kilka ciekawszych zastosowań, uporządkowanych wg. dyscyplin.

Badania operacyjne. Tutaj grafem jest harmonogram wykonywania działań. Węzłami mogą być działalności a krawędziami kolejność wykonywania działań. Krawędziami mogą być też możliwe konflikty przy wykonywaniu działań.

Biologia. Drzewa filogenetyczne pokrewieństwa gatunków (w szczególności słynna klasyfikacja Linneusza).

Chemia. Modele złożonych cząsteczek, wierzchołkami są atomy pierwiastków. Są to grafy nieskierowane.

Ekologia. Sieć powiązań żywnościowych - węzłami są gatunki a krawędziami relacja kto kogo zjada, jest to więc graf skierowany.

Fizyka. Grafy służą do modelowania różnych substancji. Wtedy wierzchołkami są cząstki atomowe a krawędziami powiązania pomiędzy jądrem i elektronem czy cząstek w atomie.

Genealogia. Tutaj słynne są drzewa genealogiczne, aczkolwiek jeśli bierzemy pod uwagę związki małżeńskie to drzewo genealogiczne nie jest drzewem w sensie podrozdziału 8.4. Węzłami w drzewie genealogicznym są osoby a krawędziami zależności typu: syn, wnuk, prawnuk, ... Drzewo można traktować jako częściowo skierowany graf (związki małżeńskie są nieskierowane).

Geografia. W kartografii problem kolorowanie map czy odwzorowania map omówione w podrozdziale 8.2.4.

Informatyka

1. Schematy blokowe i UML. Projektowanie diagramów przepływu lub schematów UML sterujących programami komputerowymi lub opisujących relacje między obiektami informatycznymi. Zazwyczaj węzły to kroki programu a krawędzie to sterowanie, zazwyczaj też są to grafy skierowane.

2. Inżynieria komputerowa. Projektowanie i analiza sieci komputerowych, tutaj węzły to komputery a krawędzie to połączenia w sieci. Zazwyczaj są to grafy nieskierowane.

3. Systemy operacyjne komputera. Drzewa priorytetów do wykonania różnych zadań. Węzłami są zadania a krawędzie to kolejność priorytetów.

4. Teoria informacji. Kody Gray'a omówione dokładnie w podrozdziale 8.2.5.

5. Zarządzanie informacją. Wyszukiwanie blokowe lub binarne rekordów omówione w podrozdziale 7.4 (szczególnie 7.4.3). Sortowanie i wyszukiwanie przez "kopcowanie".

6. Konstrukcje algorytmiczne. Wykorzystanie drzew binarnych do konstrukcji różnych słynnych algorytmów np. algorytm szybkiej transformacji Fouriera (FFT).

Inżynieria.

1. Sieci dróg. Analiza sieci dróg między miastami, ruchu na tych drogach i ustalenie które drogi należy poszerzyć, wyremontować lub dobudować. Grafy nieskierowane (gdy nie ma dróg jednokierunkowych) z wagami krawędzi.

2. Kontrola ruchu drogowego. Ustalenie, które ulice powinny być jednokierunkowe, węzłami są skrzyżowania a krawędziami ulice, graf skierowany.

3. Transmisje radiowe. Problem przyporządkowania częstości stacjom radiowym, węzłami są stacje radiowe a krawędziami możliwości zakłócania.

4. Konstrukcje drogowe. Jak skonstruować optymalnie drogi, jakie są możliwe skrzyżowania i gdzie. Węzły to skrzyżowania a krawędzie drogi.

5. Sieci elektryczne. Problemem są możliwe miejsca izolacji, węzły to komponenty sieci a krawędzie to kable.

Logika.

1. Relacje. Grafy obrazujące relacje w zbiorze węzłami są elementy zbioru a krawędziami wskazanie które elemnty są w relacji.

2. Teoria częściowo uporządkowanych zbiorów. Specjalny typ grafów tzw. diagramy Hessa.

Optymalizacja.

1. Optymalizacja sieci. Jak znaleźć drzewa spinające graf, jak najmniejszym kosztem dotrzeć do wszystkich obiektów (zajmowaliśmy się tym w podrozdziale 8.4.2).

2. Problem komiwojażera. Jak znaleźć najkrótszą drogę przechodzącą wszystkie węzły - obiekty (miasta) (zob. podrozdz. 8.2.2).

Rachunek prawdopodobieństwa. Grafy obrazują stany (wierzchołki) oraz przejścia (krawędzie) w modelach Markowa. W modelach stacjonarnych mamy do czynienia z jednym skierowanym grafem z wagami krawędzi i wierzchołków a w modelach niestacjonarnych mamy nieskończony ciąg takich grafów.

Socjologia.

1. Sieci powiązań społecznych. Tutaj węzłami są osoby a krawędziami grafu interakcje między osobami.

2. Hierarchia ważności osób. Węzłami są osoby (oddziały jakiejś firmy) a skierowane krawędzie wskazują na formę podległości.

Turystyka. Problem obejścia wszystkich obiektów turystycznych (zob. problem Eulera - podrozdz. 8.2.2. i 8.3.1).

Zarządzanie. Podziały ludzi na dwie grupy i rywalizacja między tymi grupami. Rozważa się tu grafy dwudzielne a bada poziom efektywności pracy tych grup.

Podsumowanie

- Ogromną motywacją do badania grafów są ich rozliczne zastosowania częściowo zasygnalizowane w podrozdziale 8.4.
- Pierwszym problemem, przed którym stoi informatyk jest sposób implementacji grafów. Można je implementować jako listę incydencji, listę krawędzi, macierz sąsiedztwa lub macierz incydencji. Zazwyczaj przy konstruowaniu grafów początkowym elementem jest graf pusty, bukiet lub dipol.
- Mając graf zazwyczaj chcemy przejść wszystkie krawędzie lub wierzchołki co odpowiada dostępności jednych obiektów przez inne, dlatego ważne w teorii grafów są cykle i drogi Eulera i Hamiltona.
- Jeżeli sąsiedztwo wierzchołków traktujemy jako kolizję to można rozważać jakie ładunki (kolory) trzeba nadać wierzchołkom w celu uniknięcia kolizji. Jest to problem kolorowania grafów i map.
- Również ważnym problemem jest takie rozłożenie ładunków, aby można było najszybciej dotrzeć do tych najbardziej używanych. Problem ten rozwiązuje algorytm Huffmana.

Bibliografia

[1] Biggs N., L., *Discrete Mathematics,* Oxford University Press 1989

[2] Bollobas B., *Modern Graph Theory,* Springer 1998

[3] Bryant V., Aspekty kombinatoryki, *Wydawnictwa Naukowo-Techniczne* 1977.

[4] Cormen Th., H., Leiserson Ch., E., Rivest, R., L., Stein, C., *Wprowadzenie do algorytmów,* WNT, 2004.

[5] Diestel, R., *Graph Theory,* Springer 1997 .

[6] Garnier R., Taylor L., *Discrete Mathematics for New Technology,* Second Edition, IOP Publishing Ltd 2002, University of Brighton, UK.

[7] Graham R., L., Knuth D., E., Patashnik, O., *Matematyka Konkretna*, Państwowe Wydawnictwo Naukowe, Warszawa 1996.

[8] Grossman P., *Discrete Mathematics for Computing,* Second Edition, 2002, Palgrave Macmillan Houndmills, Basingstoke, Hampshire and New York.

[9] *Handbook of discrete and combinatorial mathematics,* Rosen K., H., editor in chief, Michaels J., G., project editor ... (et al.), CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton.

[10] Hoare C. A. R., Proof of a Program. FIND *Comm. ACM,* **13,** No. 1, 1970, s.39-45.

[11] Knuth D., E., *Sztuka programowania*, Wydawnictwo Naukowo-Techniczne, Warszawa 2002, T. I-III.

[12] Krantz F., G., *Discrete Mathematics Demystified,* 2009 McGraw-Hill Companies, Inc., New York, Chicago, San Francisco.

[13] Lipski, W., *Kombinatoryka dla programistów,* Wydawnictwo Naukowo- Techniczne 2004.

[14] Lipski, W., Marek, W., *Analiza kombinatoryczna,* Państwowe Wydawnictwo Naukowe, Warszawa 1986.

[15] Lovász L., Pelikán J., Vesztergombi K., *Discrete Mathematics: Elementary and Beyond,* 2003 Springer-Verlag New York Inc.

[16] Pałka, Z., Ruciński, A., *Wykłady z kombinatoryki,* Wydawnictwa Naukowo-Techniczne, Warszawa 1998.

[17] Polya, G., Tarjan, R., E., Woods, D., R., *Notes on Introductory Combinatorics,* Birkhauser 1983

[18] Riordan, J., *An Introduction to Combinatorial Analysis,* Princeton University Press 1978

[19] Ross, K., A., Wright, Ch., R., B., *Matematyka Dyskretna,* Państwowe Wydawnictwo Naukowe, Warszawa 1996.

[20] Wilson, R., J., *Wprowadzenie do teorii grafów,* Państwowe Wydawnictwo Naukowe, Warszawa 1985.
[21] Wirth, N., *Algorytmy + struktury danych = programy,* Wydawnictwo Naukowo-Techniczne, Warszawa 2004 Wyd. 7.
[22] Materiały z wykładów Uniwersytetu Warszawskiego zamieszczone na stronie http://wazniak.mimuw.edu.pl/

Wykaz tabel

1.1. Ilość ruchów potrzebnych na przeniesienie prążków w problemie wieży z Hanoi. 8
1.2. Ilość obszarów które można uzyskać w problemie pizzy. 9
1.3. Numer pozycji na której zostanie żywa osoba w problemie Flawiusza 10
1.4. Kolejne rozwiązania uogólnionego problemu Flawiusza. 12
1.5. Wyniki metody repertuaru dla rekurencji. 14
1.6. Zastosowanie metody repertuaru do problemu cyfr. 15
1.7. Rozwiązania uogólnionego problemu cyfr 15

2.1. Liczby harmoniczne H_n. 23
2.2. Potęgi k^n i $k^{\underline{n}}$. 34

3.1. Ilość liczb całkowitych w różnego typu przedziałach. 45
3.2. Liczby $D_k^{(q)}, q = 3, 4, 5$. 51

4.1. Liczby pierwsze mniejsze od 1000. 65
4.2. Sito Eratostenesa. 69

5.1. Trójkąt Pascala. 91

6.1. Funkcje tworzące 100
6.2. Własności funkcji tworzących. 101

7.1. Rozkład sumy zmiennych losowych. 121
7.2. Działanie algorytmu - proste wstawianie. 129
7.3. Działanie algorytmu - proste wybierania 132
7.4 Działanie algorytmu sortowanie bąbelkowe 133
7.5. Działanie algorytmu - szybkie sortowanie 135

8.1. Reprezentacja grafów za pomocą macierzy sąsiedztwa 156
8.2. Reprezentacja grafu za pomocą listy incydencji 157
8.3. Reprezentacja grafu za pomocą listy krawędzi 157
8.4. Reprezentacja grafu za pomocą macierzy incydencji 158
8.5. Kody Gray'a dla $n = 5$ 176

8.6. Własności podstawowych klas grafów 177
8.7. Algorytm Kruskala 182
8.8. Algorytm Prima 183
8.9. Tabela odległości miast 183
8.10. Przykłady kodów 185
8.11. Wynik algorytmu Huffmana 188

Wykaz rysunków

1.1. Wieża w Hanoi 7
1.2. Eliminacje po pierwszym obiegu okręgu w Problemie Flawiusza. 10

2.1. Ilustracja wersji bakaliowej przemienności sumowania. 26
2.2. Graficzna ilustracja operacji na indeksach sumy. 27
2.3. Graficzna ilustracja operacji na indeksach sumy. 27

3.1. Funkcje "podłoga" i "sufit". 42
3.2. Funkcja $\{x\}$. 43
3.3. Ilustracja $x \; mod \; y$ oraz $\lfloor \frac{x}{y} \rfloor$. 53
3.4. Generator $LCG(32, 5, 17, 0)$. Wykres zależności X_n od n. . . . 55
3.5. Generator $LCG(32, 5, 17, 0)$. Wykres zależności X_n od X_{n-1}. 55

4.1. Drzewo Sterna-Brocota 70

5.1. Permutacje zbioru $\{a, b, c\}$. 83

7.1. Twierdzenie Cauchy'ego - Maclaurina. 128
7.2. Zbiory sekwencyjne 138
7.3. Zbiory sekwencyjne z 2 indeksami 139
7.4. Lista z 2 indeksami 140
7.5. Zbiory indeksowo - sekwencyjne 141
7.6. Zbiory indeksowe 143
7.7. Zbiory uporządkowane sekwencyjne przeszukiwane blokowo . 143
7.8. Zbiory uporządkowane sekwencyjne przeszukiwane binarnie . 144
7.9. Tablice mieszające ("hash table") 147

8.1. Różne typy grafów 155
8.2. Bukiet B_n 159
8.3. Dipol D_n 159
8.4. Graf kompletny K_n 159
8.5. n-ścieżka SC_n 159
8.6. n - cykl C_n 160

8.7. n-koło W_n . . . 160
8.8. Graf dwudzielny $\mathcal{G}(V_1, V_2)$ . . . 160
8.9. Pełne grafy dwudzielne, trójdzielne 161
8.10. Grafy hipersześcienne Q_n . . . 161
8.11. Grafy Kuratowskiego . . . 161
8.12. Graf Petersena . . . 162
8.13. Mapa mostów w Królewcu . . . 164
8.14. Graf mostów Królewca . . . 164
8.15. Droga i cykl Eulera . . . 165
8.16. Problemy niemieckiego komiwojażera . . . 166
8.17. Droga i cykl Hamiltona . . . 167
8.18. Izomorfizm grafów . . . 170
8.19. Izomorfizm grafów . . . 171
8.20. Problem 4 barw na przykładzie mapy USA . . . 174
8.21. Ciąg de Bruijna dla $n = 4$ . . . 178
8.22. Graf de Bruijna . . . 178
8.23. Przykłady zastosowań drzew . . . 179
8.24. Przykłady drzew . . . 180
8.25. Algorytm Kruskala . . . 181
8.26. Przykłady drzew skierowanych . . . 185
8.27. Śledzenie algorytmu Huffmana . . . 187
8.28. Drzewo $T(L)$ . . . 188

Wykaz algorytmów

1.1. Suma $S(n)$ 4
1.2. Suma $S(n)$ - drugi sposób 5
1.3. Suma $S(n)$ - trzeci sposób 5

3.1. Sortowanie przez podział i złączanie 48
3.2. Obliczanie $J_3(n)$ 50
3.3. Obliczanie $J_3(n)$ - drugi sposób 51
3.4. Konstrukcja rozwinięcia liczby n o podstawie b 58

4.1. Algorytm Euklidesa 63
4.2. Rozszerzony Algorytm Euklidesa 64
4.3. Test dzielników 67
4.4. Sito Eratostenesa 68
4.5. Testowanie liczb pierwszych Mersenne'a 69
4.6. Zamiana liczby wymiernej na reprezentację Sterna-Brocota 72
4.7. Zamiana liczby niewymiernej na reprezentację Sterna-Brocota . . 72
4.8. Algorytm RSA 77

7.1. Algorytm - Proste wstawianie 130
7.2. Algorytm - Wstawianie połówkowe 131
7.3. Algorytm - Proste wybieranie 132
7.4. Algorytm - Sortowanie bąbelkowe 134
7.5. Algorytm - Sortowanie mieszane 134
7.6. Algorytm - Quicksort 136
7.7. Tablice haszujące 148

8.1. Algorytm Fleury'ego nalezienia drogi (cyklu) Eulera 166
8.2. Algorytm Robertsa-Floresa 169
8.3. Proty algorytm testowania izomorfizmu 171
8.4. Algorytm Kruskala znajdowania minimalnego drzewa spinającego graf 181
8.5. Algorytm Prima znajdowania minimalnego drzewa spinającego graf 182
8.6. Algorytm Huffmana 186

Wykaz osób

Adleman Leonard Max (1945-) 77
Appel Kenneth Ira (1932-) 174-175
Bachmann Paul (1837-1920) 126
Bézout Étienne (1730-1783) 110
Bondy John Adrian 168
Brocot Achille (1817-1878) 70
Brooks R. L. 174
de Bruijn Nicolaas Govert (Dick) (1918-) 178
Cauchy Augustin Louis (1789-1857) 99, 127
Cayley Arthur (1821-1895) 174
Chvátal Václav Vašek (1946-) 168
Dirac Gabriel Andrew (1925-1984) 167
Erdös Paul (1913-1996) 165
Eratostenes (Eratostenes z Cyreny)(276 pne-195 pne) 68
Euklides (Euklides z Aleksandrii) (około 300 pne) 63-64, 106-107
Euler Leonhard (1707-1783) 24, 76, 163-165, 172, 177, 179, 191
Fermat Pierre de (1601 lub 1607/8-1665) 76
Fibonacci Leonardo (Leonardo Pisano Bogollo, Leonardo Bonacci) (1170-1250) 55, 101, 103, 115, 126
Flawiusz Józef (Josef ben Matatia) (37 pne-74 ne) 10, 49
Fleury (Henry?) 166
Flores Benito 169
Fourier Jean Baptiste Joseph (1768-1830) 189
Gallai Tibor (Tibor Grünwald, 1912-1992) 165
Gauss Carl Friedrich 68
Gray Frank (1954-) 161, 175, 189
Guthrie Francis (1831-1899) 174
Hadamard Jacques (1865-1963) 68
Haken Wolfgang (1928-) 174-175
Hamilton Sir William Rowan (1805-1865) 166, 168, 178, 191
Heawood Percy (1861-1955) 174
(Hesse (spolszcz.) Hasse Helmut (1898-1979) 190
Horner William George (1786-1837) 59
Huffman David Albert (1925-1999) 183, 191
Iverson Keneth E. (1920-2004) 18
Karnaugh Maurice (1924-) 176
Kempe Sir Alfred Bray (1849-1922) 174
Knuth Donald Ervin (1938-) 132
Kruskal Joseph Bernard, Jr. (1928-) 181
Kuratowski Kazimierz (1896-1980) 161, 173
Landau Edmund (1877-1938) 127
Lehmer Derrick Henry "Dick" (1905-1991) 54, 69
Leibnitz (Leibniz) Gottfried Wilhelm (1646-1716) 39
Linneusz (Linné) Carl von (1707-1778) 189
Lucas Edouard (1842-1891) 7, 69
MacLaurin Colin (1698-1746) 127
Markov Andrey (Andrei) Andreyevich (1865-1922) 190
Mersenne Marin (1588-1648) 69
de Morgan Augustus (1806-1871) 174
Morse Samuel Finley Breese (1791-1872) 187
Newton Sir Isaac (1642/1643-1727) 90, 93, 98
Ore Øystein (1899-1968) 167

Pascal Blaise (1623-1662) 91
Petersen Julius Peter Christian (1839-1910)
161, 173
Poussin Vallée (1866-1962) 68
Prim Robert Clay (1921-) 182
Rivest Ronald Linn (1947-) 77
Roberts S. M. 168
Robertson Neil 175
Sanders Daniel P. 175
Seymour Paul D. (1950-) 175
Shamir Adi (1952-) 77
Stern Moritz Abraham (1807-1894) 70
Stirling James (1692-1770) 87, 129
Taylor Brook (1685-1731) 111-112
Thomas Robin 175

Indeks

alfabet 183, 187
  alfabet Morse'a 187
algorytm 63-64, 76, 106-107, 131, 181-183, 188-189, 191
  algorytm Euklidesa 63-64, 106-107
  algorytm Huffmana 183, 188, 191
  algorytm Kruskala 181
  algorytm Prima 182
  algorytm RSA 76
  algorytmy sortowania - wstawianie połówkowe 131
  algorytmy sortowania - wybieranie 131
  algorytm szybkiej transformacji Fouriera 189
bijekcja 169
cechy podzielności liczb 73, 76
ciąg 6, 100-101, 103, 115, 126, 164, 178
  ciąg arytmetyczny 6
  ciąg de Bruijna 178
  ciąg Fibonacciego 100-101, 103, 115, 126
  ciąg geometryczny 6
  ciąg graficzny 164
cykl 154, 163, 166, 168, 177, 179, 191
  cykl Eulera 163, 177, 179, 191
  cykl Hamiltona 166, 168, 178, 191
definicja rekurencyjna 16
dekodowanie 79
diagramy Hessa 190
doświadczenie losowe 120
dodawanie rekordów 138
dokładny dzielnik 62
dolna silnia 33
dominanta 121
droga grafu 154-155, 163, 166-168, 177-178, 191
  droga Eulera 163, 177, 191
  droga Hamiltona 166, 168, 178, 191
  droga prosta grafu 155
  droga zamknięta grafu 155
drzewo 70, 180, 184, 188
  drzewo r-narne 188
  drzewo skierowane 184
  drzewo spinające graf 180
  drzewo Sterna-Brocota 70
dwumian Newtona 93, 98
entropia 184
exchange sort 133
faktoryzacja liczb całkowitych 66
funkcja 42-43, 76, 78-79, 97, 111, 146, 149-150
  funkcja ϕ Eulera 76
  funkcja część całkowita 43
  funkcja dekodująca 78-79
  funkcja kodująca 78-79
  funkcja mieszająca 146
  funkcja mieszająca - środek kwadratu 150
  funkcja mieszająca - reszta z dzielenia 149
  funkcja mieszająca - składanie brzegami 150
  funkcja mieszająca - składanie z przesuwaniem 150
  funkcja mieszająca - wycinanie 150
  funkcja mieszająca - zamiana podstawy 150
  funkcja "podłoga" 42
  funkcja "sufit" 42
  funkcja tworząca 97
  funkcja wymierna 111
generatory liczb pseudolosowych 54-55
  generator Fibonacciego 55

generator liniowy Lehmera 54
generator multyplikatywny 55
graf 154-155, 161-162, 164-166, 168, 171, 173, 177, 180-181
graf acykliczny 155, 180
graf dualny 173
graf dwudzielny 168
graf Eulera (eulerowski) 164, 177
graf hamiltonowski 166
graf Kuratowskiego 161
graf nieskierowany 154
graf Petersena 161, 173
graf planarny 171
graf prosty 155, 180
graf regularny 165
graf ściągalny 173
graf skierowany 154
graf spójny 162
graf z wagami 181
hash table 145-146
hash table - metoda łańcuchowa 146
hash table - metoda otwarta 146
homeomorfizm grafów 173
indeks chromatyczny grafu 173
indukcja matematyczna 2
indykator zdarzenia 121
insertion sort 129
izomorfizm grafów 169
klasyfikacja Linneusza 189
klucz prywatny 77-79
klucz publiczny 77-79
kod migawkowy 183
kodowanie 77-79
kody Gray'a 161, 175, 189
kolorowanie grafów planarnych 174
kolorowanie grafów 173
kolorowanie map 174
kombinacje 84-85
kombinacje bez powtórzeń 84
kombinacje z powtórzeniami 85
kongruencja 73
konwencja Iversona 18, 25, 42
korzeń drzewa 184
krawędzie wielokrotne grafu 154
kryptografia 54, 76
las 180, 182
las spinający graf 182
liść drzewa 180
liczba chromatyczna grafu 173
liczba harmoniczna 21, 23, 101, 126
liczba Mersenne'a 69
liczba pierwsza 62, 65, 68
liczba podziałów zbioru 86
liczba Stirlinga I rodzaju 87
liczba Stirlinga II rodzaju 87
lista 139
mediana 121
metody rozwiązywania rekurencji 11-12, 15, 20, 31, 101
metoda czynnika sumacyjnego 20
metoda funkcji tworzących 101
metoda repertuaru 11-12, 15, 31
metody obliczania sum 29, 32
metody różnicowe 32
metody zaburzania 29
metody Monte Carlo 54
miejsce zerowe wielomianu 110
minimalne drzewo spinające graf 181
modulnik 52
modyfikacja rekordów 138
najmniejsza wspólna wielokrotność 62
największy wspólny dzielnik liczb 62
największy wspólny dzielnik wielomianów 107
najwyższy współczynnik wielomianu 106
nieuporządkowane zbiory sekwencyjne 138
niezależne zmienne losowe 123
numeryczne obliczanie całek 54
odchylenie standardowe 122
odszyfrowywanie 79
operator mod 50, 52
operator przesunięcia 38
operator różnicowy 32
permutacje 22, 83-84
permutacje bez powtórzeń 83
permutacje z powtórzeniami 84
pętla w grafie 155
podstawowe twierdzenie arytmetyki 65-66
podziały liczby na składniki 88
prawdopodobieństwo dyskretne 120
prawdopodobieństwo waruunkowe 124

prawa operacji na sumach 22
prawo addytywności sumowania 22
prawo łączenia zbiorów indeksów sumowania 22
prawo jednorodności sumowania 22
prawo przemienności sumowania 22
prawo składania sum (telescoping) 22
problemy 7, 9-10, 103, 163, 166, 190-191
"problem cyfr" 7
problem Fibonacciego 103
problem Flawiusza 10
problem kolorowania grafów 191
problem kolorowania map 191
problem komiwojażera 166, 190
problem mostów w Królewcu 163
problem pizzy 9
proces Markowa 190
przestrzeń zdarzeń elementarnych 120
quicksort 21, 135
równanie rekurencyjne 6
różnicowanie iloczynu fukcji 38
różnicowanie ilorazu funkcji 38
różniczka (pochodna) 32
różniczkowanie iloczynu funkcji 38
różniczkowanie ilorazu funkcji 38
rdzeń (korzeń) drzewa 184
redundancja 183
rekurencja Flawiusza 49
rekurencja liniowa jednorodna rzędu drugiego 117
rozkład jednostajny 122
rozkład prawdopodobieństwa 120
rozspójnienie grafu 162
rozwiązanie równania rekurencyjnego 6
schemat Hornera 59
sekwencyjne zbiory uporządkowane przesukiwane binarnie 144
sekwencyjne zbiory uporządkowane przesukiwane blokowo 143
selection sort 131
silnia 6
sito Eratostenesa 67-68
sortowanie 129, 133-134, 189
sortowanie bąbelkowe 133
sortowanie mieszane 134
sortowanie przez "kopcowanie" 189
sortowanie przez wstawianie 129
sortowanie przez zamianę 133
spójność krawędziowa grafu 162
spójność wierzchołkowa grafu 162
splot 98
stała Eulera 24
stopień wielomianu 106
symbol Newtona 90-91, 93-94
symbolu Newtona inne pochłanianie 93
symbolu Newtona negowanie górnego indeksu 91
symbolu Newtona pochłanianie 91
symbolu Newtona reguła dodawania 91
symbolu Newtona reguła odwracania 94
symbolu Newtona reguła podwajania 93
symbolu Newtona symetria 91
symulacje 54
szybkie sortowanie "quicksort" 21, 135
szyfrowanie 79
ściana grafu 171
ściana wewnętrzna grafu 171
ściana zewnętrzna grafu 171
średnia długość słowa kodowego 183
średnia geometryczna 122
średnia harmoniczna 122
tablica Karnaugha 176
tablice mieszające 145
teoria komunikacji 183
test dzielniów 67
test Lucasa-Lehmera 69
testowanie izomorfizmu grafów 171
tożsamość Cauchy'ego 99
trójkąt Pascala 90-91
twierdzenie 39, 76, 127, 164-165, 167-168, 172-174
twierdzenie Appela-Hakena 175
twierdzenie Bondy-Chvátala 168
twierdzenie Brooks'a 174

twierdzenie Cauchy'ego-Maclaurina 127
twierdzenie Diraca 167
twierdzenie Erdös-Gallai 165
twierdzenie Eulera 76, 164-165, 172
twierdzenie Fermata 76
twierdzenie Kuratowskiego 173
twierdzenie Leibnitza dla operatora różnicowego 39
twierdzenie Leibnitza dla pochodnych 39
twierdzenie Ore 167
układ liczenia o podstawie p 56-60
ułamek prosty 111
ułamki łańcuchowe 103
usuwanie rekordów 138
waga grafu 181
wariacje 85
wariacje bez powtórzeń 85
wariacje z powtórzeniami 85
wariancja 122
wartość oczekiwana 122
wersja bakaliowa przemienności sumowania 25
wersja waniliowa przemienności sumowania 25
widmo liczby 46
wieża z Hanoi 7
wielomian moniczny (unormowany) 109
wielomian pierwszy 109
wielomian złożony 109
wielomiany względnie pierwsze 108
wierzchołek osiągalny 162
wyszukiwanie rekordów 138
względnie pierwsze liczby 68
wzór na prawdopodobieństwo całkowite 124
wzór na różnicowanie przez części 39
wzór Stirlinga 129
wzór Taylora 111-112
zasadnicze twierdzenie algebry 109
zbiór przeliczalny 120
zbiory indeksowe 142
zbiory indeksowo-sekwencyjne 140
zdarzenia elementarne 120
zmienna losowa 121